A Machine Learning Approach for Fall Detection and Daily Living Activity Recognition

Article *in* IEEE Access · March 2019

DOI: 10.1109/ACCESS.2019.2906693

2 authors:

Ali Chelli
University of South-Eastern Norway
53 PUBLICATIONS **992** CITATIONS

SEE PROFILE

Matthias Pätzold
University of Agder
394 PUBLICATIONS **7,108** CITATIONS

SEE PROFILE

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2906693, IEEE Access

A Machine Learning Approach for Fall Detection and Daily Living Activity Recognition

Ali Chelli, *Member, IEEE,* and Matthias Pätzold, *Senior Member, IEEE*

***Abstract*— The number of older people in western countries is constantly increasing. Most of them prefer to live independently and are susceptible to fall incidents. Falls often lead to serious or even fatal injuries which are the leading cause of death for elderlies. To address this problem, it is essential to develop robust fall detection systems. In this context, we develop a machine learning framework for fall detection and daily living activity recognition. We use acceleration and angular velocity data from two public databases to recognize seven different activities including falls and activities of daily living. From the acceleration and angular velocity data, we extract time and frequency domain features and provide them to a classification algorithm. In this work, we test the performance of four algorithms for classifying human activities. These algorithms are artificial neural network (ANN), K-nearest neighbors (KNN), quadratic support vector machine (QSVM), and ensemble bagged tree (EBT). New features that improve the performance of the classifier are extracted from the power spectral density of the acceleration. In a first step, only the acceleration data are used for activity recognition. Our results reveal that the KNN, ANN, QSVM, and EBT algorithms could achieve an overall accuracy of 81.2%, 87.8%, 93.2%, and 94.1%, respectively. The accuracy of fall detection reaches 97.2% and 99.1% without any false alarms for the QSVM and EBT algorithms, respectively. In a second step, we extract features from the autocorrelation function and the power spectral density of both the acceleration and the angular velocity data, which improves the classification accuracy. By using the proposed features, we could achieve an overall accuracy of 85.8%, 91.8%, 96.1%, and 97.7% for the KNN, ANN, QSVM, and EBT algorithms, respectively. The accuracy of fall detection reaches 100% for both the QSVM and EBT algorithms without any false alarm, which is the best achievable performance.**

***Index Terms*—Fall detection, activity recognition, machine learning, acceleration data, angular velocity data, feature extraction.**

I. Introduction

Advances in the diagnosis and treatment of diseases have led to an increase in life expectancy. In every country, the percentage of elderlies in the society is increasing. The World Health Organization (WHO) estimates that by 2050 the number of people over 60 years will exceed two billion [1]. With increasing age, people become more susceptible to falls. In fact, as the age increases from 65 to over 70 years, the rate of falls and fall related injuries rises from 28% to 42% according to the WHO [2]. For people over 65 years of age, fall related injuries were the leading cause of death in 2013 [3]. Moreover, fall related injuries cause a significant costs for society, making falls a major public health problem worldwide. The number of fatal falls per year is estimated by the WHO to be equal to 420,000 per year [4]. After a fall, rapid medical care can significantly reduce the potential damage from fall injuries, resulting in a higher survival rate. For this reason, fall detection systems that can detect and report falls as fast as possible are of great importance.

During the last years, the development of fall detection systems has become a hot research topic. A plethora of fall detection systems are being developed using different approaches. We can categorize the existing fall detection systems into two main classes: (i) wearable device-based systems and (ii) context-aware systems [5]. Wearable device-based systems utilize a device that is worn by the user to detect falls. These devices integrate a gyroscope and an accelerometer that can measure the acceleration and the angular velocity. The movement and activity of the user results in a temporal variation of the measured acceleration and angular velocity data, leaving different fingerprints for different activities. By analyzing the measured acceleration and angular velocity data, it is possible to determine the type of activity performed by the user. Several studies have investigated the performance of wearable device-based systems [6]–[10]. A big advantage of wearable device-based fall detection systems is that they can recognize human activity without compromising the user privacy. Widely used smartphones with built-in accelerometer and gyroscope can also be used to measure the acceleration and angular velocity as the user moves and performs various activities. The measured data can be analyzed in real time to detect falls. This fall detection approach is very attractive because it requires no new equipment and is therefore cost-effective. For wearable device-based systems, if the user forget to wear the device, it becomes impossible to monitor the person activity. This represents the major limitation of wearable device-based systems.

Context-aware systems represent the second main category of fall detection systems. These systems are based on sensors placed in the area around the user to be monitored. The sensors used for monitoring encompass floor sensors, pressure sensors, microphones, and cameras. Context-aware systems can include a single or many types of sensors which are deployed in specific areas. This makes fall detection impossible if the user leaves the monitoring area. The most common type of context-aware systems is video surveillance. To detect falls, a camera is used to capture a series of images which is subsequently processed by a classification algorithm to determine whether a fall has occurred or not [11]. The use of video surveillance for activity recognition and fall detection has been extensively investigated in the literature [12]–[17]. The main shortcoming

A. Chelli and M. Pätzold are with the Faculty of Engineering and Science, University of Agder, 4898 Grimstad, Norway (e-mails:{ali.chelli, matthias.paetzold}@uia.no).

This work is an extended version of a paper submitted to the IEEE Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2018), Bologna, Italy, September 2018.

of video surveillance systems is that they can compromise user privacy. For this reason, video surveillance is considered illegal in some countries [18]. Moreover, context-aware systems are susceptible to external events (e.g., changes in illuminance), and have high installation costs.

To evaluate the performance of fall detection systems, we need records of actual falls. However, it is very difficult to collect real-world fall data, especially for older people. Generally, we need to monitor people for several weeks to obtain records of few actual falls. In the end, these few falls are not enough to accurately evaluate the performance of the developed fall detection system. Therefore, only a few studies have adopted this approach [19]–[21]. In the absence of data of actual falls, most researchers utilize simulated falls performed by volunteers. In addition to falls, these volunteers carry out activities of daily living (ADL) to check the accuracy of the developed fall detection system and its ability to differentiate between falls and ADL.

In the literature, several activity datasets are publicly available which allow evaluating fall detection methods and assessing their performance on real-world data. An ADL database which comprises acceleration and angular velocity data are provided in [6], where a script describing the set of activities to be carried out was provided to the participants. A total of 30 participants of different genders, ages, and weights contributed to this experiment. The experiment consisted in performing ADL activities including: standing, sitting, walking, walking upstairs, walking downstairs, and lying. To collect the acceleration and angular velocity data, a smartphone was attached to the waist of each participant. On average, the total time of recording for each participant was 192 seconds. It is worth mentioning that the dataset in [6] does not include fall data, but only ADL activities. Fall related data can be found in some public databases [7]–[10]. The authors of [7] provide a fall dataset which was performed by 42 participants. Both acceleration and angular velocity data were collected during this experiment. The participants in this experiment were young healthy adults who performed planned falls. This fact makes the collected data different from that of real falls of elderly people. Due to the difficulty of gathering enough real fall data from older people, the use of mimicked fall data for testing the performance of fall detection system is a well-accepted approach by the researchers on this topic.

In this paper, we propose a machine learning framework for fall detection and activity recognition. Our first main contribution is related to the features used for fall detection. More specifically, we use the mean value of the triaxial acceleration and achieve a fall detection accuracy and precision of 96.8% and 100%, respectively. Even though, the mean value of the triaxial acceleration is not intrinsically a new feature since it was used in previous work [6] to classify ADLs, the mean value for triaxial acceleration was not utilized as a feature in the classification of falls [22], [23]. Note that by extracting only the mean value of the triaxial acceleration, we construct a feature vector of size 3. In [22], a feature vector of length 4 is used for fall detection. This resulted in a fall detection accuracy of 92% and a precision of 81%, while in [23] a feature vector of length 23 is utilized leading to a fall detection accuracy of 93.5% and a precision of 94.2%. For our solution, we use a feature vector of length 3 and achieve a fall detection accuracy and precision of 96.8% and 100%, respectively. Thus, we outperform the fall detection systems in [22] and [23] in terms of accuracy and precision by using less features.

Our second main contribution consists in proposing new features that improve the classification accuracy of ADLs. For instance, we have proposed new power spectral density (PSD) features that enhance the classification accuracy, especially, for the activities walking, walking upstairs, and walking downstairs. In the literature, several features were extracted from the PSD, such as the largest frequency value [6] and the mean frequency value [24]. However, in this paper, we extract the main peaks of the PSD and use them as a feature for activity classification. To the best of our knowledge, this feature has never been utilized before in activity classification. Moreover, we extract additional novel features, such as the peaks of the autocorrelation function (ACF), and the peaks of the cross-correlation function (CCF), which are extracted from the triaxial acceleration and the triaxial angular velocity signals. These proposed new features allow a more accurate distinction between different activities.

In this work, we combine the fall and ADL data from the datasets provided in [7] and [6]. These real-world data are then utilized to evaluate the performance of the proposed machine learning framework in human activity recognition. The acceleration and angular velocity signals are divided into buffers of 2.56 s length. From each buffer, we extract a feature vector of length 66, in a first step. To improve the accuracy of the classification, more features are extracted from each buffer, such that the length of the feature vector increases to 328. Note that the lengths of the considered feature vectors (66 and 328) are smaller than the number of features used in existing baseline solutions. We utilize 70% of the data to train the classifier, while 30% of the data are used to test the trained classifier. For a feature vector of length 66, we achieve a similar performance compared to existing solutions [24], while for a feature vector of length 328, our approach outperforms existing solutions.

In this paper, we assess the performance of four different classification algorithms, namely, the artificial neural network (ANN), K-nearest neighbors (KNN), quadratic support vector machine (QSVM), and ensemble bagged tree (EBT). In a first step, only the acceleration data are used for feature extraction. A feature vector of length 66 is built and provided as input to the classification algorithm. Our results reveal that the KNN algorithm has the worst performance with an overall accuracy of 81.2%. The EBT algorithm has the best performance with an overall accuracy of 94.1%. The ANN and the QSVM algorithms achieve an overall accuracy of 87.8% and 93.2%, respectively. The accuracy of fall detection reaches 97.2% and 99.1% for the QSVM and EBT algorithms, respectively, without any false alarm. In a second step, we extract features from both the acceleration and the angular velocity data and construct a feature vector of length 328. This increase in the number of features improves the performance of the four classification algorithms. The KNN, the ANN, the QSVM, and the EBT algorithms achieve an overall accuracy of 85.8%, 91.8%, 96.1%, and 97.7%, respectively. Moreover, the accuracy of fall detection reaches 100% for both QSVM and EBT without any false alarm, which is the best achievable performance.

The remainder of the paper is organized as follows. Section II describes the machine learning framework, the different blocks in this framework, and their roles. We discuss the time domain and frequency domain features in Section III. In Section IV, we assess the accuracy and the precision of our proposed solution using first the features from the acceleration data only and then using features from both the acceleration and angular velocity data. Finally, Section V offers concluding remarks.

II. Framework Description

Our objective is to determine the user's activity based on the measured acceleration and angular velocity data. In this section, we provide an overview of the framework used for classifying ADLs as well as fall events and explain the activity recognition strategy. Fig. 1 illustrates the activity recognition framework which encompasses: (i) the input acceleration and angular velocity data obtained from the smartphone, (ii) the feature extraction block, and (iii) the classification algorithm. In the following, we discuss each component of this framework.

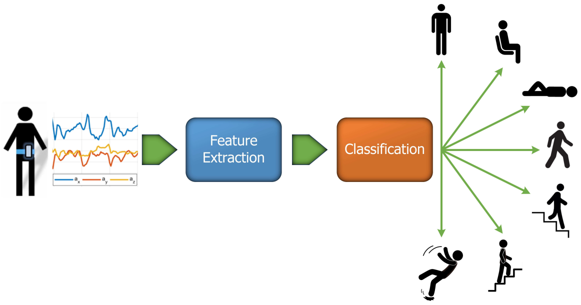


Fig. 1. Activity recognition framework.

A. Data Description and Preprocessing

The triaxial angular velocity and acceleration data are obtained from two public databases. The first database in [6] comprises six types of activities: walking, walking upstairs, walking downstairs, sitting, standing, and lying. A total of 30 participants were involved in this experiment. A smartphone was attached to the waist of the participants to collect acceleration and angular velocity data. The sampling frequency of the collected data was 50 Hz. The data have then been divided into buffers of 2.56 s length with 50% overlap. Each data buffer is labeled with the corresponding actual activity using the ground truth and contains both the triaxial acceleration and the triaxial angular velocity of a specific participant. In addition to the ADL data set, we acquired acceleration and angular velocity data for fall events from the public database in [7]. Our aim is to develop a framework that uses the acceleration and angular velocity data to classify seven types of activities: falling, walking, walking upstairs, walking downstairs, sitting, standing, and lying. Since the data obtained from the two databases in [6] and [7] is provided as an input to the classification algorithm, this data must be homogenous. The data from [7] are organized into buffers of length 2.56 s to make it consistent with the data from the first database [6]. Moreover, we select the fall data from 30 participants given in [7]. The collected triaxial acceleration data can be written as

$$a_x(t) = a_x^g(t) + a_x^b(t) \tag{1}$$

$$a_y(t) = a_y^g(t) + a_y^b(t) \tag{2}$$

$$a_z(t) = a_z^g(t) + a_z^b(t) \tag{3}$$

where $a_x(t)$, $a_y(t)$, and $a_z(t)$ refer to the acceleration data measured along the x-axis, y-axis, and z-axis, respectively. The acceleration $a_x(t)$ in (1) is expressed as a sum of two terms: (i) $a_x^g(t)$ which stands for the gravity contribution to the acceleration along the x-axis and (ii) $a_x^b(t)$ which denotes the body movement contribution to the acceleration along the x-axis. Similarly, the accelerations $a_y(t)$ and $a_z(t)$ are written as a sum of two terms as shown in (2) and (3). Henceforth, the terms $a_i(t)$ $(i = x, y, z)$ and $a_i^b(t)$ $(i = x, y, z)$ are referred to as the total acceleration and the body acceleration, respectively. Since the body acceleration $a_i^b(t)$ $(i = x, y, z)$ reflects the impact of the body movement on the measured acceleration, the use of the body acceleration for activity recognition should intuitively yield a better classification accuracy. Hence, we must filter out the gravity contribution to the measured total acceleration to obtain the body acceleration. Generally, the contribution of gravity to the acceleration varies slowly which implies that the gravity component in the frequency domain occurs at frequencies near to 0 Hz. This is as opposed to the contribution of the body movement which occurs at frequencies larger than 0 Hz. Therefore, we can eliminate the gravity contribution by applying a high-pass filter to the total acceleration $a_i(t)$ $(i = x, y, z)$. To this end, we use a Chebyshev filter of Type II [25] with a stopband attenuation of 60 dB and stopband frequency of 0.4 Hz. It is worth mentioning that Type II Chebyshev filters are sharper than Butterworth filters which allows filtering out the gravity contribution [25]. Moreover, Type II Chebyshev filters can extract the body acceleration from the total acceleration with negligible distortions, since Type II Chebyshev filters have no ripples for frequencies larger than the passband frequency [25].

From the total acceleration signal provided by the smartphone, we can obtain other signals, such as the triaxial body acceleration signal and the magnitude of the body acceleration signal. By increasing the number of signals from which features are extracted, we can improve the accuracy of human activity recognition. The triaxial body acceleration signal $a_i^b(t)$ $(i = x, y, z)$ is extracted from the total acceleration signal by applying a high-pass filter to the acceleration data $a_i(t)$ $(i = x, y, z)$. The magnitude of the body acceleration can be expressed as

$$\|\boldsymbol{a}^b(t)\| = \sqrt{[a_x^b(t)]^2 + [a_y^b(t)]^2 + [a_z^b(t)]^2}. \tag{4}$$

Beside the triaxial total acceleration, the smartphone is equipped with a gyroscope that can measure the angular velocity. The angular velocities around the x-, y-, and z-axes are denoted as $\omega_x(t)$, $\omega_y(t)$, and $\omega_z(t)$, respectively. The unit of the measured angular velocities is radians per second $(\mathrm{rad/s})$. By integrating the angular velocity with respect to time, we can obtain the time-variant angular position. The angular rotation around the x-, y-, and z-axes are referred to as the pitch, the roll, and the yaw, respectively. Since the triaxial angular velocities are not affected by the gravity, these data reflect the impact of the body movement. Thus, the gyroscope data can be used directly without any filtering. Moreover, we compute the magnitude of the angular velocity and use it to extract additional features to improve the classification accuracy of the proposed framework. The signals used for features extraction are listed in Table I.

TABLE I
THE SIGNALS UTILIZED FOR FEATURE EXTRACTION.

Signal Name	Notation
Triaxial total acceleration	$a_i(t)\ (i = x, y, z)$
Triaxial body acceleration	$a_i^b(t)\ (i = x, y, z)$
Magnitude of body acceleration	$\|\boldsymbol{a}^b(t)\|$
Triaxial angular velocity	$\omega_i(t)\ (i = x, y, z)$
Magnitude of angular velocity	$\|\boldsymbol{\omega}(t)\|$

B. Feature Extraction

This section offers an overview of the concept of feature extraction and highlights its importance in obtaining an accurate classification. The acceleration and angular velocity signals are provided as input to the feature extraction block as shown in Fig. 1. Afterwards, the output of the feature extraction block is used by the classification algorithm to recognize human activities.

It is worth noting that if we directly provide the classification algorithm with raw acceleration and angular velocity data, the classification algorithm will fail to distinguish different types of activities and the classification accuracy would be very poor. In classification problems, the aim is to distinguish between different classes of activities. A good feature must achieve this objective. For instance, a good feature can have a specific range of values for each activity, and these value ranges do not overlap. In this case, by knowing the value of the considered feature, we can find out to which range it belongs and consequently recognize the type of the performed activity. Moreover, a good feature must be general enough such that it allows identifying the activity associated with new data. These are two criteria that must be fulfilled by a good feature. Note that raw data does not fulfill any of these criteria.

Additionally, the raw data is generally contaminated with noise and artifacts which makes it very difficult for the classifier to find any pattern in the data. Moreover, if the raw data is used, the dimensionality of the feature vector becomes huge and makes the processing of that feature vector complex and time consuming. Feature extraction helps to reduce the dimensionality of the problem and decreases therefore its complexity. By selecting the right features, we reduce the complexity of activity recognition and improve the classification accuracy. This renders feature extraction a corner stone in achieving a high classification accuracy with reasonable complexity.

The task of feature extraction consists in finding a finite set of measures that captures quantitative descriptions and enables differentiating between various classes of activity. Typical features include statistical quantities extracted from the acceleration signal, such as the mean value, the standard deviation, and higher order moments [26], [27].

In the following, we consider a simple example to explain how feature extraction can help to determine the type of activity performed by the user. We assume for simplicity that the collected acceleration data pertain to two activities: lying[1] and standing. Our objective is to determine the user activity based on the observed acceleration data. By carefully studying the acceleration data, we find certain properties in the data that can be used to recognize the performed activity. For instance, for lying, the acceleration data $a_z(t)$ has a mean value that is close to 0 $\mathrm{m/s^2}$. In contrast, for standing the mean value of the acceleration $a_z(t)$ is around 10 $\mathrm{m/s^2}$. We assume now that we receive a new acceleration data buffer which could be measured either while the user was lying or standing. Our task is to recognize the activity that was performed by the user when this acceleration buffer was recorded. There are two possible outcomes: (i) the user activity is lying or (ii) the user activity is standing. A simple way to recognize the user activity consists of evaluating the mean value of the acceleration $a_z(t)$. Then, this mean value is provided to the classification algorithm. If the mean value of the acceleration data $a_z(t)$ is close to 0 $\mathrm{m/s^2}$, the classifier would decide that the performed activity is lying. Otherwise, if the mean value of the acceleration data $a_z(t)$ is close to 10 $\mathrm{m/s^2}$, the classifier decides that the performed activity is standing.

In this example, a distinction was made between two activities using the collected acceleration data. To achieve this goal, we have used a single feature. This feature is the mean value of the acceleration $a_z(t)$. In this paper, we raise a much more complicated problem. Our aim is to achieve a good classification accuracy for seven types of activities. Therefore, we need to extract a large number of features. In Section III, we discuss in detail all the features used in our proposed solution to achieve a high classification accuracy.

C. Classification Algorithm

The objective of the classification algorithm is to recognize the user activity based on the acceleration and gyroscope data. We use a supervised learning approach to achieve this objective. As a first step, the algorithm is exposed to a large set of labeled data[2], the so-called training data. Based on the training data, the classification algorithm can tune its internal parameters to reduce the misclassification rate as much as possible. After the training phase, the classification accuracy

[1] The acceleration data for the activity lying has been recorded while the participant is laying down and in the phase right before laying down.

[2] The class of the data is given to the classification algorithm.

of the algorithm is assessed using a new set of data, called the test data.

First, we recall that the data are organized in buffers of length 2.56 s. Each of these buffers is labeled with an activity identity (ID) indicating to which class the data buffer belongs. The activity IDs are numbered from 1 to 7. The activity IDs 1, 2, 3, 4, 5, 6, and 7 correspond to walking, walking upstairs, walking downstairs, sitting, standing, lying, and falling, respectively. For example, if the data buffer has an activity ID equal to 4, this implies that the data buffer was recorded while the participant is sitting. The data buffer provided to the feature extraction block contains raw acceleration and angular velocity data. The feature extraction block extracts the set of features described in Section III. After computing the value of each feature for the considered data buffer, these features are stacked in a vector, known as the feature vector. This vector is provided to the classification algorithm which must recognize the type of activity performed by the user, while the data buffer was recorded. To achieve a good classification accuracy, the classification algorithm must first be trained to learn the underlying pattern of each activity. During the training phase, the classification algorithm is exposed to labeled data to optimize its internal parameters such that the classification error is minimized. Subsequently, we can assess the performance of the trained algorithm using the test data. Once a new buffer is received, the corresponding feature vector is determined and provided to the classifier. This latter computes the likelihood that this buffer belongs to one of the seven possible activity classes. The algorithm then declares that the buffer belongs to the activity with the highest likelihood score. For example, if Class 5 has the highest score for a given buffer, then the algorithm would declare that the user was standing. To find out whether the decision of the algorithm is right or wrong, we compare it with the ground truth (the labeled data). This process is repeated for each buffer in the test data. By combining all the results, we generate a confusion matrix that shows the accuracy and the precision of the classifier for each activity. In this paper, we evaluate the performance of four classification algorithms to recognize human activity based on the collected acceleration and angular velocity data. These four classification algorithms are the ANN, KNN, QSVM, and EBT algorithm. Principles and background information about the ANN, KNN, QSVM, and EBT algorithms can be found in [28]–[30].

III. Feature Extraction

The raw acceleration and angular velocity signals could be utilized as inputs to the classification algorithm. However, in this case, the accuracy of the activity recognition would be very poor. To solve this problem, it is important to extract a set of features from the acceleration and the angular velocity signals. These features should have different value ranges for different activities. During the training phase, the classification algorithm is exposed to a large set of labeled data. For each activity, the classification algorithm has to learn the value range of each feature. When a new acceleration and angular velocity signal is received, the features are extracted and stored in a feature vector which is provided to the classification algorithm. The trained classification algorithm maps this feature vector to one of the seven activity classes. The accuracy of this classification depends strongly on the extracted features.

In this section, we discuss the features which are extracted from the signals presented in Table I. We explain the methods used to obtain these features and highlight their impact on improving the classification accuracy. The set of features can be divided into two main categories: time domain features and frequency domain features of the acceleration and the gyroscope data. The time domain features include the mean value, the root mean square, the main maxima and minima, and the peaks of the ACF and the peaks of the CCF of the signals given in Table I. The frequency domain features include the value and location of the main peaks of the PSD and the energy in different frequency bands of the signals listed in Table I.

The sample mean of the total acceleration is our first statistical feature. By exploring the mean value of the total acceleration of different activities, we find that for lying the mean value of $a_x(t)$ is equal to $0\ \mathrm{m/s^2}$, while for activities where the human body is in a vertical position, such as standing and walking, the mean value of the acceleration $a_x(t)$ is equal to $10\ \mathrm{m/s^2}$. Using this property, we can differentiate lying from other activities. The histogram of the accelerations $a_x(t)$ and $a_z(t)$ pertaining to the activities standing and lying is illustrated in Fig. 2. It can be seen from this figure that the mean value of $a_x(t)$ equals $10\ \mathrm{m/s^2}$ for standing and $0\ \mathrm{m/s^2}$ for lying. On the other hand, the mean value of the acceleration $a_z(t)$ equals $0\ \mathrm{m/s^2}$ and $5\ \mathrm{m/s^2}$ for standing and lying, respectively.

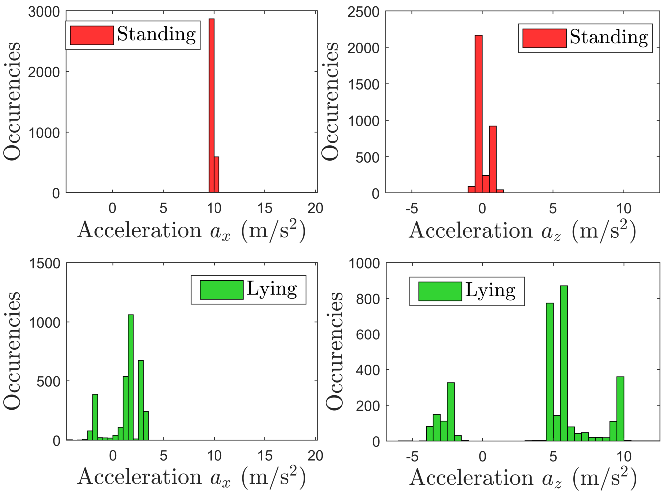


Fig. 2. Histogram of the accelerations $a_x(t)$ and $a_z(t)$ for the activities standing and lying.

Note that the orientation of the accelerometer axes when lying is different compared to activities with vertical body posture, such as standing and walking. Thus, depending on the body posture (vertical or horizontal), we observe different mean values of $a_x(t)$. We recall that the contribution of the body acceleration is negligible in comparison to the gravity pertaining to the activities standing and lying. For standing, the gravitational field contribution equals 10 $\mathrm{m/s^2}$ along the x-axis of the accelerometer[3], while the contribution of the body acceleration is zero along all axes. As opposed to standing, the impact of the gravitational field for lying is equal to 0 $\mathrm{m/s^2}$ along the x- and y-axes of the accelerometer and 10 $\mathrm{m/s^2}$ along the z-axis. Fig. 2 shows that the mean value of $a_z(t)$ is 5 $\mathrm{m/s^2}$ for lying. This is because the collected acceleration data are recorded while the user is performing lying and during lying which makes the mean value of $a_z(t)$ smaller than 10 $\mathrm{m/s^2}$. But even with this error, an accurate classification of the lying activity is obtained by extracting the mean value of the total acceleration. Note that this feature has not been considered in previous studies.

In addition to the total acceleration, we evaluate the sample mean of the other signals provided in Table I to obtain additional features. More specifically, we compute the mean values of the magnitude of the body acceleration $\|\boldsymbol{a}^b(t)\|$, the triaxial angular velocity $\omega_i(t)$ $(i = x, y, z)$, and the magnitude of the angular velocity $\|\boldsymbol{\omega}(t)\|$.

The second feature that we extract is the root mean square (RMS) of the body acceleration. The RMS is also known as the quadratic mean. The body acceleration $a_i^b(t)$ $(i = x, y, z)$ is obtained by applying a high-pass filter to the total acceleration $a_i(t)$ $(i = x, y, z)$. This filtering removes the contribution of the gravitational field. The RMS of the body acceleration can be expressed as

$$a_i^{b,\mathrm{RMS}} = \sqrt{\frac{1}{T}\int_0^T \left[a_i^b(t)\right]^2 dt} \quad \text{for} \quad i = x, y, z \tag{5}$$

where T is the length of the buffer which is equal to 2.56 s. Beside the RMS of the body acceleration, we incorporate additional features obtained by computing the RMS of the angular velocity $\omega_i(t)$ $(i = x, y, z)$, the RMS of the magnitude of the body acceleration $\|\boldsymbol{a}^b(t)\|$, and the RMS of the magnitude of the angular velocity $\|\boldsymbol{\omega}(t)\|$.

In Fig. 3, we illustrate the histogram of the RMS of the body acceleration for the activities sitting and walking downstairs. From this figure, it can be deduced that the RMS of the body acceleration $a_x^b(t)$ for sitting is less than 0.2, while for walking downstairs the RMS of $a_x^b(t)$ is larger than 0.2. In contrast, the RMS of the body acceleration $a_y^b(t)$ is smaller than 0.1 for the activity sitting and mostly larger than 0.1 for the activity walking downstairs.

By examining the RMS for the activity standing, which is a static activity similar to sitting, we notice that the RMS of $a_x^b(t)$ and $a_y^b(t)$ is less than 0.1 and 0.2, respectively. On the other hand, for the dynamic activities, such as walking, walking upstairs, walking downstairs, and falling the RMS of $a_x^b(t)$ and $a_y^b(t)$ is larger than 0.1 and 0.2, respectively. Thus, this feature allows differentiating between static and dynamic activities.

The third feature is the main maxima and minima of the triaxial body acceleration $a_i^b(t)$ $(i = x, y, z)$. We apply a Savitzky-Golay filter [31] to the body acceleration to smooth it and reduce the impact of noise. The Savitzky-Golay smoothing method reduces the noise while preserving the underlying pattern and the peaks in the data. By exploring the histograms of different activities, we find that the ranges of the acceleration values vary. For instance, the acceleration mean value for the activities walking and standing is the same, but the dynamic range of the accelerations is different. Consequently, by extracting the main maxima and minima of the acceleration, we can reduce the misclassification rate for walking and standing. Note that this feature improves the classification accuracy of all activities.

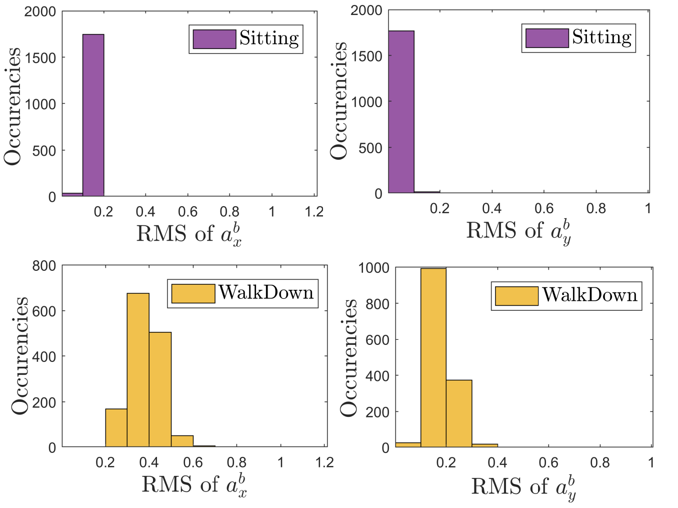


Fig. 3. Histogram of the RMS of the body accelerations $a_x^b(t)$ and $a_y^b(t)$ for the activities sitting and walking downstairs.

It is worth mentioning that the above features allow distinguishing between activities that exhibit very different acceleration patterns, i.e., activities with different acceleration mean values and variances. Nevertheless, for activities with similar statistical properties, the classification based on the above features would result in poor accuracy. For example, we notice that the activities walking, walking downstairs, and walking upstairs have similar mean and variance. If we would use only the above features to classify the activities walking, walking upstairs, and walking downstairs, we would find a misclassification error of more than 15%. We must investigate how these acceleration signals vary over time to discriminate acceleration signals associated with these activities. More specifically, we must measure the rate of oscillations of the acceleration. Actually, people tend to move slower when walking upstairs compared to walking downstairs which results in a higher rate of oscillations if the person is walking downstairs. By extracting the peaks of the PSD, we can obtain a quantitative description of the rate and shape of the oscillations of the acceleration signal.

Our forth feature quantifies the rate of change and shape of the oscillation of the body acceleration signal $a_i^b(t)$ $(i = x, y, z)$. This feature is extracted from the PSD of the acceleration, which can be obtained as follows. First, we compute the ACF $R_{a_i^b}(\tau)$ of the body acceleration $a_i^b(t)$ $(i = x, y, z)$

[3] If the person is standing, the x-axis of the accelerometer corresponds to the z-axis of the earth-centered coordinate system.

as

$$R_{a_i^b}(\tau) = \frac{1}{2T}\int_{-T}^{T} a_i^b(t+\tau)\left[a_i^b(t)\right]^* d\tau. \qquad (6)$$

The PSD $S_{a_i^b}(f)$ of the body acceleration $a_i^b(t)$ $(i = x, y, z)$ can be obtained by applying the Fourier transform to the ACF $R_{a_i^b}(\tau)$ as

$$S_{a_i^b}(f) = F\left\{R_{a_i^b}(\tau)\right\} = \int_{-\infty}^{\infty} R_{a_i^b}(\tau)e^{-j2\pi f\tau}d\tau \quad \text{for } i = x, y, z. \qquad (7)$$

From the PSD $S_{a_i^b}(f)$, we extract the location and the value of the main PSD peaks. Our hypothesis is that these PSD peaks capture well the time variation in the acceleration signal and allow identifying the fundamental frequency and main harmonic frequencies embedded in the acceleration signal. Thus, with the help of the PSD feature, we can distinguish between different activities, since each activity leads to different time variations in the acceleration signal as well as to different shapes and rates of oscillations. In the following, we provide arguments supporting our hypothesis that the proposed PSD feature describes well the observed time variations in the acceleration signal. Thus, the PSD feature allows improving the classification accuracy for different activities.

Our analysis of the PSD $S_{a_i^b}(f)$ of the triaxial body acceleration reveals that this PSD is narrowband and can generally be considered equal to zero outside the frequency interval $[f_{\min}, f_{\max}]$[4]. Using this property, we can write the inverse Fourier transform of the PSD as

$$\begin{aligned} F^{-1}\left\{S_{a_i^b}(f)\right\} &= \int_{-\infty}^{\infty} S_{a_i^b}(f)e^{j2\pi f\tau}df \quad \text{for } i = x, y, z. \\ &= \int_{f_{\min}}^{f_{\max}} S_{a_i^b}(f)e^{j2\pi f\tau}df \\ &\approx \sum_{n=1}^{N} S_{a_i^b}(f_n)e^{j2\pi f_n\tau}\Delta f \qquad (8) \end{aligned}$$

where the approximation in (8) is obtained using [32, Eq. (7)] and $\Delta f = (f_{\max} - f_{\min})/N$. Note that as $N \to \infty$ the approximation in (8) becomes an equality. In (8), the number of terms in the sum can be reduced from N to P $(P \ll N)$ by selecting the terms with the P largest weights $S_{a_i^b}(f_p)$ $(p = 1, 2, \ldots, P)$. Thus, we can write

$$\sum_{n=1}^{N} S_{a_i^b}(f_n)e^{j2\pi f_n\tau}\Delta f \approx \sum_{p=1}^{P} S_{a_i^b}(f_p)e^{j2\pi f_p\tau}\Delta f. \qquad (9)$$

Note that the P components on the right-hand side of (9) coincide with the P peaks of the PSD, which we extract as a feature to distinguish the activities walking, walking upstairs, and walking downstairs.

On the other hand, the ACF $R_{a_i^b}(\tau)$ can be expressed as the inverse Fourier transform of the PSD $S_{a_i^b}(f)$. Using (8) and (9), we can write

$$R_{a_i^b}(\tau) = F^{-1}\left\{S_{a_i^b}(f)\right\} \approx \sum_{p=1}^{P} S_{a_i^b}(f_p)e^{j2\pi f_p\tau}\Delta f. \qquad (10)$$

Therefore, the ACF $R_{a_i^b}(\tau)$ of the triaxial acceleration can be approximated by P harmonics with weights $S_{a_i^b}(f_p)$ and frequency f_p. The ACF $R_{a_i^b}(\tau)$ of the triaxial acceleration contains information pertaining to the time variation of the acceleration signal. Moreover, the ACF enables finding repeating patterns in the acceleration signal as well as identifying the fundamental frequency and main harmonic frequencies embedded in the acceleration signal. Thus, by extracting the location (f_p) and the value $(S_{a_i^b}(f_p))$ of P peaks of the PSD $S_{a_i^b}(f)$, we capture quantitative information on the time variation of the acceleration signal as shown in (10).

In Fig. 4, we illustrate the PSDs $S_{a_x^b}(f)$ and $S_{a_y^b}(f)$ of the body accelerations $a_x^b(t)$ and $a_y^b(t)$ for the activities walking and walking upstairs. From this figure, we see that most of the information is confined to the range from 0 to 10 Hz. The peak locations and values hold useful information on the shape and rate of the signal oscillations in the time domain. From the PSD curves $S_{a_x^b}(f)$ and $S_{a_y^b}(f)$, we observe a fundamental frequency f_0 around 1 Hz and a number of harmonics at positions that are multiples of f_0. The relative amplitudes of the spectral peaks are closely related to the shape of oscillation of the signal, whereas the spacing between the spectral peaks indicates the rate of oscillation of the signal.

For the activity walking upstairs, it can be seen from the PSDs $S_{a_x^b}(f)$ and $S_{a_y^b}(f)$ that the spectral peaks are closer together and pushed to the left compared to the spectral peaks for the activity walking. This means that the rate of oscillation for walking is higher than that of the activity walking upstairs. Besides, for the activity walking upstairs, the amplitude of the peaks to the right of the fundamental frequency f_0 decreases quickly. This implies that the shape of the oscillation for walking upstairs is smoother compared to walking. This can be explained by Newton's second law of motion which states that the sum of forces is equal to the mass times the acceleration [33]. When people are walking upstairs, the gravity impact makes the body acceleration smaller and the shape of its oscillations smoother compared to walking on a flat surface, where the gravity has almost no impact on the body acceleration.

The classification accuracy for walking, walking downstairs, and walking upstairs is improved by using the spectral peaks features. We recall that the use of other features, such as the mean, the RMS, and the maxima does not yield an accurate classification for these activities. The proposed frequency domain feature enhances the accuracy of the classification algorithm, especially for the activities walking, walking downstairs, and walking upstairs. Beside the spectral peaks of the body acceleration $a_i^b(t)$ $(i = x, y, z)$, we extract as well the spectral peaks of the angular velocity $\omega_i(t)$ $(i = x, y, z)$, the spectral peaks of the magnitude of the body acceleration $\|\boldsymbol{a}^b(t)\|$, and the spectral peaks of the magnitude of the angular velocity $\|\boldsymbol{\omega}(t)\|$.

The fifth feature is extracted from the ACF of the body acceleration $a_i^b(t)$ $(i = x, y, z)$. More specifically, we estimate the values and the location of the first maximum and the

[4] Typically $f_{\min} = 0$ Hz and $f_{\max} = 10$ Hz.

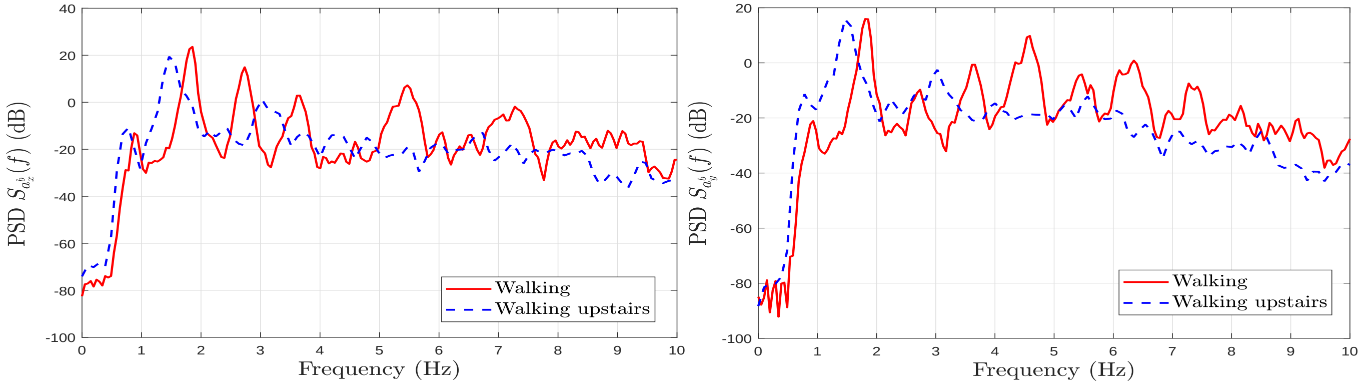


Fig. 4. PSDs $S_{a_x^b}(f)$ and $S_{a_y^b}(f)$ of the body accelerations $a_x^b(t)$ and $a_y^b(t)$ pertaining to the activities walking and walking upstairs.

second peak of the body acceleration ACF. These features contain information pertaining to the shape and rate of change of the oscillations of the acceleration signal. Such features can improve the classification of activities that have similar statistical properties (i.e., similar mean values and variances) but have a different rate and shape of oscillations. Additionally, we extract similar features from the ACF of the angular velocity $\omega_i(t)$ $(i = x, y, z)$, the ACF of the body acceleration $\|\boldsymbol{a}^b(t)\|$, and the ACF of the magnitude of the angular velocity $\|\boldsymbol{\omega}(t)\|$.

Our sixth feature quantifies the energy in different frequency bands of the triaxial body acceleration $a_i^b(t)$ $(i = x, y, z)$. To extract this feature, we first obtain the PSD of the body acceleration. Then, we divide the frequency spectrum into 10 bands and evaluate the energy confined in each frequency band. To improve the classification accuracy, we extract as well the energy in different bands of the triaxial angular velocity $\omega_i(t)$ $(i = x, y, z)$, the magnitude of the body acceleration $\|\boldsymbol{a}^b(t)\|$, and the magnitude of the angular velocity $\|\boldsymbol{\omega}(t)\|$.

The seventh feature is extracted from the cross-correlation function (CCF) between the body acceleration on different axes. More specifically, we estimate the values and the locations of the first three peaks of the CCF. These peaks provide information about the level of resemblance between the body acceleration measured on different axes. We determine the CCF between the body acceleration signal pairs ($a_x^b(t)$, $a_y^b(t)$), ($a_x^b(t)$, $a_z^b(t)$), and ($a_y^b(t)$, $a_z^b(t)$). Then, we extract the locations and values of the first three peaks of these CCFs.

IV. Experimental Results

In this section, we assess the performance of the proposed activity recognition framework. The dataset is divided into two random independent sets: the training set and the test set. We use 70% of the data for training and 30% for testing. In our investigation, we evaluate the performance of the ANN, the KNN, the QSVM, and the EBT classification algorithms.

A. *Classification Based on the Acceleration Signal*

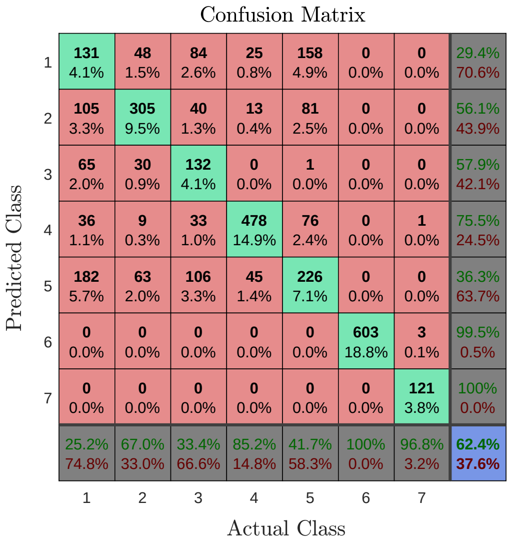
In a first step, we extract features only from the triaxial total acceleration $a_i(t)$ $(i = x, y, z)$ and the triaxial body acceleration $a_i^b(t)$ $(i = x, y, z)$. To emphasize the importance of the proposed features for improving the accuracy of the classification, we arrange the features into three subsets: Subset A, Subset B, and Subset C. Subset A comprises the mean value of the triaxial total acceleration which is referred to as the first feature in Section III. Subset B includes the features from Subset A augmented with the peaks extracted from the PSD and the ACF of the body acceleration which represent the fourth and the fifth features as described in Section III. Finally, Subset C encompasses the features from Subset B in addition to the RMS and the main maxima and minima of the body acceleration. The feature vector of Subset C has a length of 66 and contains features extracted only from the acceleration data.

We consider an ANN classification algorithm with one hidden layer, which comprises 25 nodes. The performance of this ANN algorithm is assessed using the features of Subset A. The obtained results are provided in the confusion matrix in Fig. 5. In this figure, the diagonal cells show the number and the percentage of correct classifications by the trained ANN algorithm. For instance, in 131 cases the classifier correctly predicts the walking activity. These 131 cases represent 4.1% of the 3200 buffers that are being classified during the test phase by the trained ANN classifier. Similarly, the ANN algorithm successfully predicted the class of 305, 132, 478, 226, 603, and 121 data buffers as pertaining to the activities walking upstairs, walking downstairs, sitting, standing, and falling, respectively.

By observing a given column of the confusion matrix in Fig. 5, it is possible to know the accuracy of the algorithm for a given class[5]. For example, the first column shows the results associated with the activity walking. The first row of Column 1 contains the number 131, which implies that in 131 cases the activity walking was successfully recognized by the ANN algorithm. The value in the second row of Column 1 indicates that in 105 cases the algorithm misclassified the activity walking as walking upstairs. Similarly, the value in row j $(j = 2, \ldots, 7)$ of Column 1 indicates the number of

[5] Throughout the paper the words class and activity are used interchangeably. Class i corresponds to the activity with ID $i (i = 1, 2, \ldots, 7)$. The number $i (i = 1, 2, \ldots, 7)$ located to the left of the confusion matrix in Fig. 5 indicates that the predicted class is Class i.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2906693, IEEE Access



Confusion Matrix

	1	2	3	4	5	6	7	
1	**131** 4.1%	**48** 1.5%	**84** 2.6%	**25** 0.8%	**158** 4.9%	**0** 0.0%	**0** 0.0%	29.4% 70.6%
2	**105** 3.3%	**305** 9.5%	**40** 1.3%	**13** 0.4%	**81** 2.5%	**0** 0.0%	**0** 0.0%	56.1% 43.9%
3	**65** 2.0%	**30** 0.9%	**132** 4.1%	**0** 0.0%	**1** 0.0%	**0** 0.0%	**0** 0.0%	57.9% 42.1%
4	**36** 1.1%	**9** 0.3%	**33** 1.0%	**478** 14.9%	**76** 2.4%	**0** 0.0%	**1** 0.0%	75.5% 24.5%
5	**182** 5.7%	**63** 2.0%	**106** 3.3%	**45** 1.4%	**226** 7.1%	**0** 0.0%	**0** 0.0%	36.3% 63.7%
6	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**603** 18.8%	**3** 0.1%	99.5% 0.5%
7	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**121** 3.8%	100% 0.0%
	25.2% 74.8%	67.0% 33.0%	33.4% 66.6%	85.2% 14.8%	41.7% 58.3%	100% 0.0%	96.8% 3.2%	**62.4%** **37.6%**

Predicted Class
Actual Class

Fig. 5. Confusion matrix of the ANN algorithm obtained using the features from Subset A.

cases for which the activity walking was misclassified as the activity with the ID j [6]. The accuracy for activity one indicates the percentage of successful classification for the activity walking. This accuracy is obtained by dividing two numbers: (i) the first quantity is the number of buffers pertaining to the activity walking and that are correctly classified[7] and (ii) the second quantity is the total number of buffers pertaining to the activity walking[8]. For the activity walking, the classification accuracy equals 25.2% as shown in the first column of Row 8. For example, considering the falling events which are represented over the seventh column. In total there are 125 falls in the considered test data. In 121 cases, the fall events are correctly recognized by the classifier which yields an accuracy of 96.8%. The classifier fails to recognize fall events in 4 cases which means that 3.2% of the classifications for fall events are incorrect. The classification accuracy of activity j is provided in Column j $(j = 1, \ldots, 7)$ of Row 8. The classification accuracy for the activities walking upstairs, walking downstairs, sitting, standing, lying, and falling are equal to 67%, 33.4%, 85.2%, 41.7%, 100%, and 96.8%, respectively. Overall the ANN classifier was able to successfully predict the user activity in 62.4% of the cases.

By looking at a given row of the confusion matrix in Fig. 5, we can evaluate the prediction precision for a given class. For instance, let us consider the fourth row which corresponds to sitting. The activity sitting is correctly predicted in 478 cases and wrongly predicted in 155 cases, which implies a precision of 75.5% for the predictions of the activity sitting. The activities walking, walking upstairs, walking downstairs, standing and falling are misclassified as sitting, in 36, 9, 33, 76, and 1 case, respectively. Out of 633 sitting predictions, 155 predictions are wrong which represents 24.5%. The classification precision of activity j is provided in row $j (j = 1, \ldots, 7)$ of Column 8. The classification precision for the activities walking, walking upstairs, walking downstairs, sitting, standing, lying, and falling are equal to 29.4%, 56.1%, 57.9%, 75.5%, 36.3%, 99.5%, and 100%.

It is worth mentioning that the accuracy and the precision of the classification have different meaning. The accuracy focuses on the actual activity and indicates the percentage of successful classifications out of the actual buffers belonging to a given class. In contrast, the precision of a classification focuses on the predicted activity and quantifies the percentage of successful classification out of the buffers predicted to belong to a certain activity.

Fig. 5 shows that the classifier recognizes with high accuracy the activities lying and falling. These two activities are almost never confused with the remaining five activities. As shown by the acceleration histograms in Fig. 2, one can visually differentiate between the activity lying and other activities based on the range of values of the acceleration along the x- and z-axis. This explains the high accuracy in recognizing the activity lying, which reaches 100%. Similarly, Fig. 10 compares the histograms of the mean value of the acceleration for falling and standing. Using Fig. 10, one can distinguish falls from non-falls based on the range of the mean value of the acceleration. This clarifies the high fall detection accuracy, which reaches 96.8%.
On the other hand, Fig. 5 demonstrates that the classifier confuses the activities walking, walking upstairs, walking downstairs, sitting, and standing, since all of them have similar patterns for the histogram of the mean value of the acceleration.[9] For instance, the algorithm misclassifies walking as sitting in 182 cases and as walking upstairs in 105 cases.

In Table II, we provide the confusion matrix of the binary classification problem, where we classify the data into fall and non-fall classes. The non-fall class includes the activities walking, walking upstairs, walking downstairs, sitting, standing, and lying. Table II is obtained when using the ANN classifier with the features from Subset A. From the binary confusion matrix, we can compute the false negative (FN) and false positive (FP) as well as the FN rate and FP rate for fall detection. From Table II, we see that the number of FP equals 0 and the number of FN is 4. The FP rate can be computed as

$$\begin{aligned}\text{FP Rate} &= \frac{\text{FP}}{\text{Number of actual non–falls}} = \frac{\text{FP}}{\text{FP} + \text{TN}} \\ &= \frac{0}{3075} = 0\%. \end{aligned} \tag{11}$$

Since the FP rate is equal to 0%, this implies that if the classifier is given a non-fall event, it never recognizes it as a fall. Thus, the fall detection system has zero false alarm. The FN rate can be computed by dividing the number of FN

[6] The activity IDs 1, 2, 3, 4, 5, 6, and 7 correspond to walking, walking upstairs, walking downstairs, sitting, standing, lying, and falling, respectively.

[7] This number, which is shown in the first row of the first column, is equal to 131 [see the confusion matrix in Fig. 5].

[8] By summing the numbers in Column 1 located in Rows 1-7, we get the total number of buffers that actually belong to the activity walking.

[9] We recall that the only feature used by the classifier in Fig. 5 is the mean value of the triaxial acceleration.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2906693, IEEE Access

by the number of actual falls, i.e.,

$$\begin{aligned} \text{FN Rate} &= \frac{\text{FN}}{\text{Number of actual falls}} = \frac{\text{FN}}{\text{TP} + \text{FN}} \\ &= \frac{4}{125} = 3.2\%. \end{aligned} \qquad (12)$$

The FN rate indicates the percentage of undetected falls by the system. It is desirable that the fall detection system has a very low FN rate.

Using Table II, we can compute the accuracy and precision for fall detection as follows

$$\text{Accuracy} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{121}{121 + 4} = 96.8\% \qquad (13)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{121}{121} = 100\%. \qquad (14)$$

TABLE II

Confusion matrix of the binary classification problem of the ANN algorithm obtained using the features from Subset A.

	Actual Non-Fall	Actual Fall
Predicted Non-Fall	3075 (**TN**)	4 (**FN**)
Predicted Fall	0 (**FP**)	121 (**TP**)

To demonstrate the importance of the proposed features in improving the classification accuracy, we assess the performance of the ANN classifier using the features of the Subsets A, B, and C. Table III shows the classification accuracy results of the ANN algorithm. It can be noticed from this table that as the set of features becomes larger, the overall accuracy of the classifier is improved. For example, if we consider the activity walking, we see that if we use the features from Subset A, we achieve a poor classification accuracy of 25.2%. If we use the features of Subset B instead of Subset A as an input to the ANN algorithm, the classification accuracy for walking is enhanced by more than 60%. Note that Subset B encompasses the features from Subset A augmented with the peaks of the PSD and the ACF. These additional features hold information pertaining to the shape and rate of the oscillations of the acceleration signals which improves the classification accuracy for most activities. Moreover, if we use the features of Subset B instead of Subset A, we observe that the classification accuracy is improved by 11.8%, 39.2%, and 43.7% for the activities walking upstairs, walking downstairs, and standing, respectively. From Table III, we observe that by using the features of Subset C, the classification accuracy is enhanced furthermore. The ANN algorithm achieves an accuracy of 96.8% and 100% for the activities falling and lying when using the features of Subset C.

TABLE III

Accuracy of the ANN classifier for various activities and different feature subsets.

	Accuracy %							
Features	Wal.	Up.	Dow.	Sit.	Sta.	Ly.	Fal.	Overall
Subset A	25.2	67	33.4	85.2	41.7	100	96.8	62.4
Subset B	85.6	78.8	72.6	80.8	85.4	100	93.8	85.1
Subset C	88.4	81.3	84.1	84.8	83	100	96.8	87.8

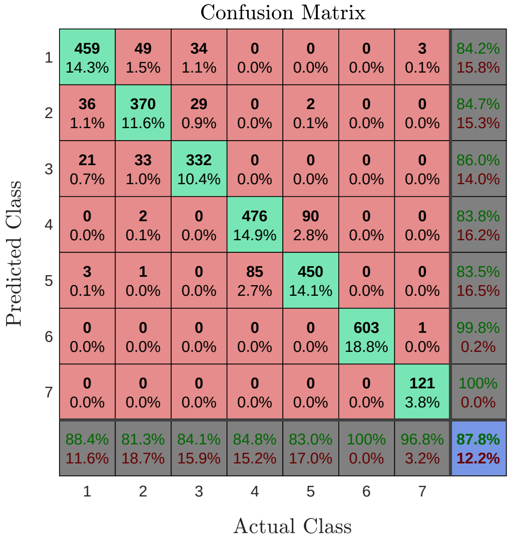
The precision of the ANN algorithm achieved with the features of the Subsets A, B, and C is provided in Table IV. This table shows that the precision of the predicted falls reaches 100% regardless of whether we use the features of Subset A, B, or C. This implies that there are no false alarms and that all fall events detected by the algorithm are real falls. On the contrary, as the number of features increases, the false alarm rate for walking decreases. For example, if we use the features of Subset B instead of Subset A, the classification precision for walking is enhanced by 48.2%. Additionally, if we use the features of Subset C instead of Subset A, the classification precision is improved by 28.6%, 28.1%, 8.3%, and 47.2%, respectively, for the activities walking upstairs, walking downstairs, sitting, and standing.

TABLE IV

Precision of the ANN classifier for various activities and different feature subsets.

	Precision %						
Features	Wal.	Up.	Dow.	Sit.	Sta.	Ly.	Fal.
Subset A	29.4	56.1	57.9	75.5	36.3	99.5	100
Subset B	77.6	79.9	80.8	83.7	82	99.8	100
Subset C	84.2	84.7	86	83.8	83.5	99.8	100

The confusion matrix of the ANN classifier obtained by using the features of Subset C is illustrated in Fig. 6. The diagonal cells of the confusion matrix provide the number and the percentage of correct classifications. For example, the classifier correctly predicts fall events in 121 cases. These 121 cases represent 3.8% of the total number of buffers which are classified by the ANN algorithm during the test phase. From the remaining diagonal cells of the confusion matrix in Fig. 6, we can conclude that the trained ANN algorithm successfully predicts the class of the activities walking, walking upstairs, walking downstairs, sitting, and standing in 459, 370, 332, 476, 450, and 603 cases, respectively.



	1	2	3	4	5	6	7	
1	**459** 14.3%	**49** 1.5%	**34** 1.1%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**3** 0.1%	84.2% 15.8%
2	**36** 1.1%	**370** 11.6%	**29** 0.9%	**0** 0.0%	**2** 0.1%	**0** 0.0%	**0** 0.0%	84.7% 15.3%
3	**21** 0.7%	**33** 1.0%	**332** 10.4%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	86.0% 14.0%
4	**0** 0.0%	**2** 0.1%	**0** 0.0%	**476** 14.9%	**90** 2.8%	**0** 0.0%	**0** 0.0%	83.8% 16.2%
5	**3** 0.1%	**1** 0.0%	**0** 0.0%	**85** 2.7%	**450** 14.1%	**0** 0.0%	**0** 0.0%	83.5% 16.5%
6	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**603** 18.8%	**1** 0.0%	99.8% 0.2%
7	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**121** 3.8%	100% 0.0%
	88.4% 11.6%	81.3% 18.7%	84.1% 15.9%	84.8% 15.2%	83.0% 17.0%	100% 0.0%	96.8% 3.2%	**87.8%** **12.2%**

Fig. 6. Confusion matrix of the ANN algorithm obtained using the features from Subset C.

To assess the accuracy of the algorithm for a given activity, we must observe the corresponding column for that activity in the confusion matrix in Fig. 6. For instance, let us consider the first column of the confusion matrix which pertain to the activity walking. In the test data, there are 519 walking buffers. In 459 cases, the walking activity is successfully recognized by the algorithm which implies an accuracy of 88.4%. The activity walking is misclassified in 60 cases which represent 11.6% of the actual walking buffers. Overall the ANN algorithm successfully predicts the user activity in 87.8% of the cases.

To determine the algorithm precision for a given activity, we must look at the row corresponding to that activity in the confusion matrix in Fig. 6. For example, let us consider the fifth row which pertain to the activity standing. In 450 cases, the activity standing is predicted correctly. This implies that the prediction precision for standing is 83.5%. The activities sitting, walking upstairs, and walking are misclassified as standing in 85, 1, and 3 cases, respectively. Out of 539 standing predictions, 89 predictions are wrong which represents 16.5%.

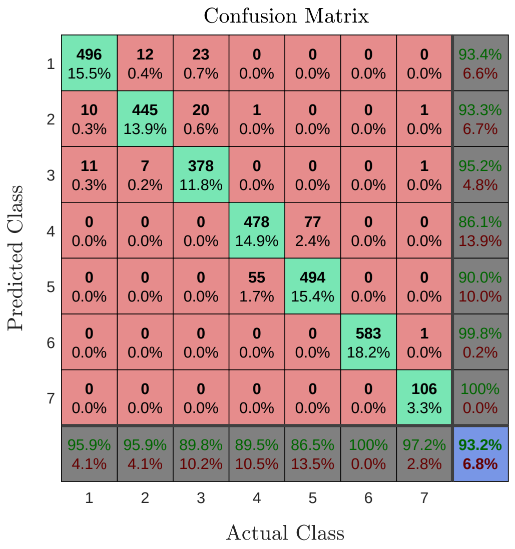
From Fig. 6, we observe that the classifier clearly distinguishes lying and falling from the other activities. For the remaining five activities, the classifier confuses standing and sitting together, since these two activities are static. Besides, the classifier does not differentiate well the dynamic activities walking, walking upstairs, and walking downstairs. However, the misclassification rate among dynamic activities drops significantly by using the features from Subset B instead of those from Subset A, as shown in Table III. This demonstrates that the PSD features allow achieving a higher accuracy in recognizing dynamic activities, since each of these activities has its own rate and shape of oscillations as discussed in Section III. Note that the classifier rarely misclassifies dynamic activities as static and vice versa. For example, the number of misclassifications for the activity walking as standing drops from 182 to 3 by using the feature Subset C instead of Subset A. This reveals that the features in Subset C allow distinguishing static and dynamic activities.

In Table V, we provide the confusion matrix of the binary classification problem, where we classify the data into fall and non-fall classes. Table V is obtained when using the ANN classifier with the features from Subset C. From Table V, we see that the number of FP equals 0 and the number of FN is 4. The FP rate and FN rate can be computed using (11) and (12), which results in 0% and 3.2%, respectively. Utilizing (13) and (14), we can compute the accuracy and precision of fall detection which are equal to 96.8% and 100%, respectively.

TABLE V
CONFUSION MATRIX OF THE BINARY CLASSIFICATION PROBLEM OF THE ANN ALGORITHM OBTAINED USING THE FEATURES FROM SUBSET C.

	Actual Non-Fall	Actual Fall
Predicted Non-Fall	3075 (**TN**)	4 (**FN**)
Predicted Fall	0 (**FP**)	121 (**TP**)

The confusion matrix for the QSVM classifier is provided in Fig. 7. Comparing the confusion matrices of the ANN with that of the QSVM algorithm, we observe that the QSVM algorithm outperforms the ANN algorithm in terms of the overall accuracy by 5.4%. Moreover, the QSVM algorithm has better accuracy and precision for most activities compared to the ANN algorithm. If the QSVM algorithm is utilized instead of the ANN algorithm, the prediction precision is improved by 9.2%, 8.6%, 9.2%, 2.3%, and 6.5%, respectively, for the activities walking, walking upstairs, walking downstairs, sitting, and standing. Additionally, for the activities walking, walking upstairs, walking downstairs, sitting, standing, and falling, the classification accuracy is improved by 7.5%, 14.6%, 5.7%, 4.7%, 3.5%, and 0.4%, respectively, if we use the QSVM algorithm instead of the ANN algorithm.



	1	2	3	4	5	6	7	
1	**496** 15.5%	**12** 0.4%	**23** 0.7%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	93.4% 6.6%
2	**10** 0.3%	**445** 13.9%	**20** 0.6%	**1** 0.0%	**0** 0.0%	**0** 0.0%	**1** 0.0%	93.3% 6.7%
3	**11** 0.3%	**7** 0.2%	**378** 11.8%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**1** 0.0%	95.2% 4.8%
4	**0** 0.0%	**0** 0.0%	**0** 0.0%	**478** 14.9%	**77** 2.4%	**0** 0.0%	**0** 0.0%	86.1% 13.9%
5	**0** 0.0%	**0** 0.0%	**0** 0.0%	**55** 1.7%	**494** 15.4%	**0** 0.0%	**0** 0.0%	90.0% 10.0%
6	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**583** 18.2%	**1** 0.0%	99.8% 0.2%
7	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**106** 3.3%	100% 0.0%
	95.9% 4.1%	95.9% 4.1%	89.8% 10.2%	89.5% 10.5%	86.5% 13.5%	100% 0.0%	97.2% 2.8%	**93.2%** **6.8%**

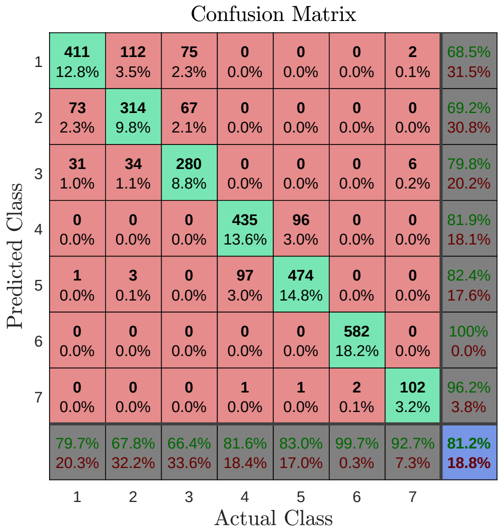
Fig. 7. Confusion matrix of the QSVM algorithm obtained using the features from Subset C.

In Table VI, we provide the confusion matrix of the binary classification problem, where we classify the data into fall and non-fall classes. Table VI is obtained when using the QSVM algorithm with the features from Subset C. From Table VI, we see that the number of FP equals 0 and the number of FN is 3. The FP rate is equal to 0%, while the FN rate is 2.75%. The accuracy and precision for fall detection are equal to 97.25% and 100%, respectively.

TABLE VI
CONFUSION MATRIX OF THE BINARY CLASSIFICATION PROBLEM OF THE QSVM ALGORITHM OBTAINED USING THE FEATURES FROM SUBSET C.

	Actual Non-Fall	Actual Fall
Predicted Non-Fall	3090 (**TN**)	3 (**FN**)
Predicted Fall	0 (**FP**)	106 (**TP**)

In Figs. 8 and 9, we provide the confusion matrices for the KNN and the EBT algorithms, respectively. These results are obtained using the features from Subset C. From Fig. 8, it can be noticed that the KNN algorithm has the worst overall accuracy compared to the ANN, QSVM, and EBT algorithms, while the EBT algorithm has the best performance in terms of the overall accuracy. By using the EBT algorithm instead of the KNN algorithm, we can improve the overall accuracy by 12.9%. The EBT algorithm significantly improves the classification accuracy of most activities compared to the KNN algorithm. In particular, the EBT algorithm enhances the classification accuracy for the activities walking, walking upstairs, walking downstairs, sitting, standing, and falling by 14.1%, 25.7%, 22.9%, 11.7%, 9.3%, and 6.4%, respectively. Moreover, the use of the EBT algorithm improves the precision of the predictions and significantly reduces false alarms compared to the KNN algorithm. More specifically, if the EBT algorithm is utilized instead of the KNN algorithm, the precision of the prediction for the activities walking, walking upstairs, walking downstairs, sitting, standing, and falling is improved by 22.7%, 23.5%, 13.7%, 10%, 11.2%, and 3.8%, respectively.



	1	2	3	4	5	6	7	
1	411 12.8%	112 3.5%	75 2.3%	0 0.0%	0 0.0%	0 0.0%	2 0.1%	68.5% 31.5%
2	73 2.3%	314 9.8%	67 2.1%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	69.2% 30.8%
3	31 1.0%	34 1.1%	280 8.8%	0 0.0%	0 0.0%	0 0.0%	6 0.2%	79.8% 20.2%
4	0 0.0%	0 0.0%	0 0.0%	435 13.6%	96 3.0%	0 0.0%	0 0.0%	81.9% 18.1%
5	1 0.0%	3 0.1%	0 0.0%	97 3.0%	474 14.8%	0 0.0%	0 0.0%	82.4% 17.6%
6	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	582 18.2%	0 0.0%	100% 0.0%
7	0 0.0%	0 0.0%	0 0.0%	1 0.0%	1 0.0%	2 0.1%	102 3.2%	96.2% 3.8%
	79.7% 20.3%	67.8% 32.2%	66.4% 33.6%	81.6% 18.4%	83.0% 17.0%	99.7% 0.3%	92.7% 7.3%	81.2% 18.8%

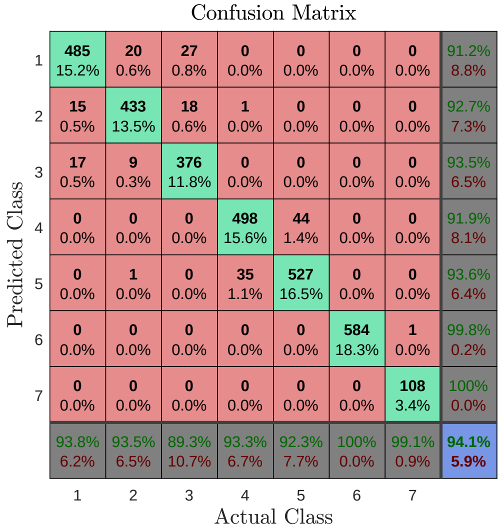
Fig. 8. Confusion matrix of the KNN algorithm obtained using the features from Subset C.

In Table VII, we provide the confusion matrix of the binary classification problem. Table VII is obtained when using the KNN algorithm with the features from Subset C. From Table VII, we see that the number of FP equals 4 and the number of FN is 8. The FP rate is equal to 3.77%, while the FN rate is 7.27%. The accuracy and precision of fall detection are equal to 96.23% and 100%, respectively.

TABLE VII
CONFUSION MATRIX OF THE BINARY CLASSIFICATION PROBLEM OF THE KNN ALGORITHM OBTAINED USING THE FEATURES FROM SUBSET C.

	Actual Non-Fall	Actual Fall
Predicted Non-Fall	3085 (**TN**)	8 (**FN**)
Predicted Fall	4 (**FP**)	102 (**TP**)

Table VIII illustrates the confusion matrix of the binary classification problem resulting from using the EBT algorithm with the features from Subset C. Table VIII shows that the number of FP and FN are equal to 0 and 1, respectively. The FN rate equals 0.91%, whereas the FP rate is 0%. For fall detection, the accuracy and precision reach 99.09% and 100%, respectively.



	1	2	3	4	5	6	7	
1	485 15.2%	20 0.6%	27 0.8%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	91.2% 8.8%
2	15 0.5%	433 13.5%	18 0.6%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	92.7% 7.3%
3	17 0.5%	9 0.3%	376 11.8%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	93.5% 6.5%
4	0 0.0%	0 0.0%	0 0.0%	498 15.6%	44 1.4%	0 0.0%	0 0.0%	91.9% 8.1%
5	0 0.0%	1 0.0%	0 0.0%	35 1.1%	527 16.5%	0 0.0%	0 0.0%	93.6% 6.4%
6	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	584 18.3%	1 0.0%	99.8% 0.2%
7	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	108 3.4%	100% 0.0%
	93.8% 6.2%	93.5% 6.5%	89.3% 10.7%	93.3% 6.7%	92.3% 7.7%	100% 0.0%	99.1% 0.9%	94.1% 5.9%

Fig. 9. Confusion matrix of the EBT algorithm obtained using the features from Subset C.

TABLE VIII
CONFUSION MATRIX OF THE BINARY CLASSIFICATION PROBLEM OF THE EBT ALGORITHM OBTAINED USING THE FEATURES FROM SUBSET C.

	Actual Non-Fall	Actual Fall
Predicted Non-Fall	3090 (**TN**)	1 (**FN**)
Predicted Fall	0 (**FP**)	108 (**TP**)

B. Comparison

In this work, we obtained the acceleration data for ADL activities and falls from two different databases. This fact makes it difficult to compare our results to existing work in the literature. In [24], the authors use the support vector machine (SVM) algorithm to classify six ADL activities using the same acceleration data that we use in this paper. Therefore, we can roughly compare our results to those obtained in [24]. The classification accuracy in [24] for the activities walking, walking upstairs, walking downstairs, standing, sitting, and lying are equal to 95.6%, 69.8%, 83.2%, 93%, 96.4%, and 100%, respectively, while the overall accuracy reaches 89.3%. In our case, we achieve a better overall accuracy of 93.2%, if we use the QSVM algorithm and the features extracted only from the acceleration signal. Note that in our case we classify seven different activities compared to six activities in [24]. Our solution improves the classification accuracy for the activities walking, walking upstairs, and walking downstairs by 0.3%, 26.1%, and 6.6% compared to the method proposed in [24]. On the other hand, the solution in [24] outperforms our method for the classification of the activities standing and sitting by 3.5% and 9.9%, respectively.

Wearable-based fall detection systems use either thresholding or machine learning algorithms to detect falls. Threshold-based algorithms have low complexity and can be easily implemented on wearable devices. Their major drawback is that they produce a high number of false alarms [23]. In fact, threshold-based algorithms conclude that a fall has occurred if the magnitude of the acceleration vector exceeds a certain value. Such a simple algorithm confuses falls with activities that yield a large acceleration value, such as walking downstairs [34]. The use of machine learning algorithms to detect falls is quite popular due to their high accuracy, which is achieved at a larger computational cost compared to thresholding algorithms.

Many studies have investigated the performance of different fall detection algorithms using acceleration data [22], [23]. We compare the performance of our proposed machine learning framework to [22], [23]. The choice of these two papers as a benchmark is motivated by two reasons. First, the fall data used to assess the performance of our fall detection solution is the same as the fall data used in [22], [23], which makes this comparison fair. Second, the solutions proposed in [22], [23] have a high fall detection accuracy and precision.

In [22], the acceleration and angular velocity data are collected by two sensors attached to the participants' chests and thighs. The vector magnitude of the acceleration and angular velocity obtained from the two sensors are computed and stacked in a feature vector of length 4. A decision tree algorithm is used to classify fall and non-fall activities. The performance in terms of fall detection reaches an accuracy of 92% and a precision of 81%.

In [23], the authors built a binary classifier which can distinguish between fall and non-fall events. A feature vector of length 23 was provided to the classifier to decide if a fall has occurred or not. The performance of three classification algorithms was evaluated, namely, decision tree, logistic regression, and multilayer perceptron. The best performance in [23] was achieved with the multilayer perceptron classifier which has a fall detection accuracy of 93.5% and a precision of 94.2%.

In our proposed solution, by just using the acceleration fall data obtained from the sensor attached to the chest, we achieve a fall detection accuracy and precision of 96.8% and 100%, respectively[10], by utilizing only a feature vector of length 3 (features from Subset A). This feature vector contains the mean value of the acceleration along the x-axis, y-axis, and z-axis. Thus, by using less data and a smaller size feature vector, we are able to outperform the fall detection systems proposed in [22] and [23]. In our framework, the precision and accuracy of fall detection are further improved by increasing the size of the feature vector. For instance, using a feature vector of length 66, the EBT algorithm has a fall detection accuracy of 99.1% and a precision of 100%.

In the following, we explain why we achieve a better fall detection accuracy than [22] and [23], even though we use less features. In [22], the authors utilize as features the vector magnitude of the acceleration and the angular velocity. In [23], the authors extract features from the magnitude of the acceleration signal, such as the minimum, maximum, mean, variance, and signal magnitude area. However, in both [22] and [23] there are no features extracted from the triaxial acceleration signal, but the features are extracted from the magnitude of the acceleration data. By computing the magnitude of the acceleration signal, we combine the acceleration data from the x-axis, y-axis, and z-axis into a single value, but we lose important information on the orientation of the acceleration vector $\boldsymbol{a}(t)$ that helps to recognize falls. Next, we explain this idea in more detail.

In Fig. 10, we illustrate the histogram of the mean value of the accelerations $a_x(t)$ and $a_z(t)$ for the activities standing and falling. For standing, the mean value of the acceleration $a_x(t)$ is mainly confined in the interval $[9\ \mathrm{m/s^2}, 10\ \mathrm{m/s^2}]$, while for falling, the mean value of $a_x(t)$ is mostly between 0 and 7 $\mathrm{m/s^2}$. This mismatch between the histograms of the mean value of the acceleration $a_x(t)$ for standing and falling allows distinguishing between these two activities using a threshold. Note that for the activities where the body posture is vertical (i.e., walking, walking upstairs, walking downstairs, sitting, and standing), the mean value of the acceleration $a_x(t)$ is approximately equal to the gravity contribution, which is measured along the x-axis of the accelerometer[11], and equal to $10\ \mathrm{m/s^2}$. Thus, using the mean value of the acceleration $a_x(t)$, we can easily distinguish between falling and all the activities where the body posture is vertical.

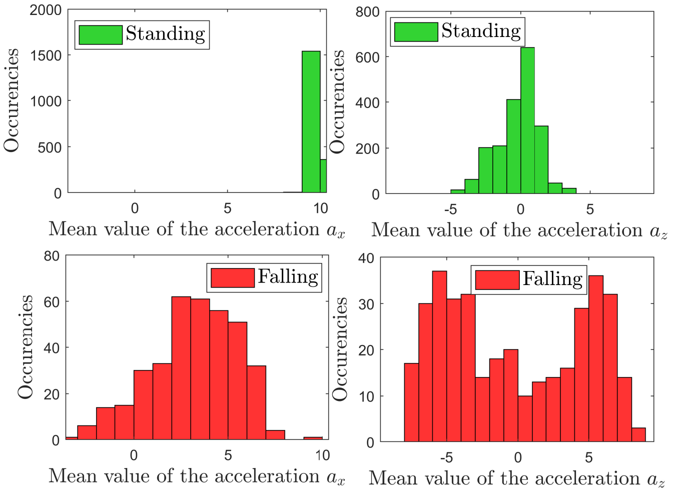


Fig. 10. Histogram of the mean value of the accelerations $a_x(t)$ and $a_z(t)$ for the activities standing and falling.

A fall comprises three main stages: (i) the pre-fall, (ii) the fall, and (iii) the post-fall. In the pre-fall stage, the person is generally walking and has a vertical body posture, while for the post-fall stage the person is usually lying on the ground. During the fall stage, the body posture changes from

[10] See the confusion matrix in Fig. 5.

[11] For the activities with vertical body posture, the x-axis of the accelerometer corresponds to the z-axis of the earth-centered coordinate system.

vertical to horizontal. Since the smartphone is attached to the person's body, the orientation of the axes of the smartphone's accelerometer changes during the fall. For activities with a vertical body posture, such as standing, the x-axis of the accelerometer coincides with the z-axis of the earth. Therefore, for a vertical body posture, the gravity contribution equals to $10\ \mathrm{m/s^2}$ along the x-axis and $0\ \mathrm{m/s^2}$ along the y- and z-axes.[12] In contrast, for activities with horizontal body posture, such as lying, the z-axis of the accelerometer coincides with the z-axis of the earth-centered coordinate system. As a result, the gravity contribution equals to $10\ \mathrm{m/s^2}$ along the z-axis and $0\ \mathrm{m/s^2}$ along the x- and y-axes. However, for falling, since the accelerometer axes orientation keeps changing during the fall, the contribution of the gravity is non-zero along both the x- and z-axes of the accelerometer as shown in Fig. 10.

By comparing the histograms of the mean value of the acceleration $a_z(t)$ for the activities standing and falling in Fig. 10, we observe that for standing the mean value of the acceleration $a_z(t)$ is mainly concentrated around $0\ \mathrm{m/s^2}$, whereas for falling the mean value of the acceleration $a_z(t)$ is generally located in the interval $[-8\ \mathrm{m/s^2}, 8\ \mathrm{m/s^2}]$. This difference in the histograms of the mean value of $a_z(t)$ for standing and falling allows the classifier to distinguish these two activities. Hence, using the mean value of the triaxial acceleration, we can improve the fall detection accuracy as it is shown in Fig. 5, where a fall detection accuracy of 96.8% is achieved using just a feature vector of size 3.

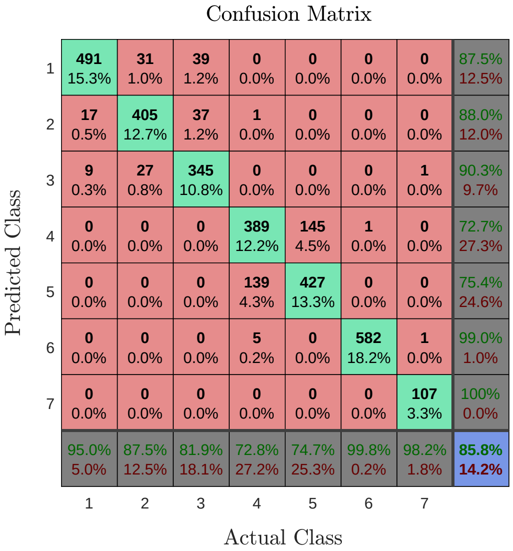
C. Classification Based on Acceleration and Angular Velocity Signals

In this section, we utilize all the features extracted from the acceleration and the angular velocity signals. We extract the mean value of the total triaxial acceleration $a_i(t)$ $(i = x, y, z)$, the triaxial angular velocity $\omega_i(t)$ $(i = x, y, z)$, the magnitude of the body acceleration $\|\boldsymbol{a}^b(t)\|$, and the magnitude of the angular velocity $\|\boldsymbol{\omega}(t)\|$. Additionally, we extract the RMS, the ACF peaks, the PSD peaks, and the energy in different frequency bands from the triaxial body acceleration $a_i^b(t)$ $(i = x, y, z)$, the triaxial angular velocity $\omega_i(t)$ $(i = x, y, z)$, the magnitude of the body acceleration $\|\boldsymbol{a}^b(t)\|$, and the magnitude of the angular velocity $\|\boldsymbol{\omega}(t)\|$. Finally, we extract supplementary features from the triaxial body acceleration $a_i^b(t)$ $(i = x, y, z)$, such as the cross-correlation peaks and the main maxima and minima. Using all these features, we construct a feature vector of length 328.

It is important to mention that machine learning algorithms have a nested and non-linear structure, which makes it difficult to understand how classifiers can achieve a high recognition accuracy. Most researchers in this field use machine learning methods as a black box [35], [36]. The interpretability of the decision made by machine learning algorithms is still an open research question [35], [36]. When we are faced with a problem where the size of the feature vector is small, we can to a certain extent interpret the obtained results. However, as the dimensionality of the feature vector increases, the interpretability of the obtained results becomes highly difficult. Since the size of the feature vector in this section is large, we cannot explain the reasons behind the obtained classification results.

In the following, we assess the accuracy of four classification algorithms using all the extracted features (i.e., the used feature vector has a length of 328). These four classification algorithms are the KNN, the ANN, the QSVM, and the EBT algorithm. The data are randomly divided into a training set and test set. The training data and the test data represent 70% and 30%, respectively, of the total data.

In Fig. 11, we provide the confusion matrix of the KNN algorithm obtained using all the features extracted from the acceleration and the angular velocity. The KNN algorithm achieves an overall accuracy of 85.8%. We recall that using the features from Subset C, we achieve an overall accuracy of 81.2%. This implies that by increasing the size of the feature vector from 66 to 328, while using the KNN algorithm, we can improve the classification accuracy by 4.6%. From Fig. 11, we see that for the KNN algorithm the classification accuracy for the activities walking, walking upstairs, walking downstairs, sitting, standing, lying, and falling are equal to 95%, 87.5%, 81.9%, 72.8%, 74.7%, 99.8%, and 98.2%, respectively. On the other hand, the KNN algorithm achieves a precision of 87.5%, 88%, 90.3%, 72.7%, 75.4%, 99%, and 100% for the activities walking, walking upstairs, walking downstairs, sitting, standing, lying, and falling, respectively.



	1	2	3	4	5	6	7	
1	**491** 15.3%	**31** 1.0%	**39** 1.2%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	87.5% 12.5%
2	**17** 0.5%	**405** 12.7%	**37** 1.2%	**1** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	88.0% 12.0%
3	**9** 0.3%	**27** 0.8%	**345** 10.8%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**1** 0.0%	90.3% 9.7%
4	**0** 0.0%	**0** 0.0%	**0** 0.0%	**389** 12.2%	**145** 4.5%	**1** 0.0%	**0** 0.0%	72.7% 27.3%
5	**0** 0.0%	**0** 0.0%	**0** 0.0%	**139** 4.3%	**427** 13.3%	**0** 0.0%	**0** 0.0%	75.4% 24.6%
6	**0** 0.0%	**0** 0.0%	**0** 0.0%	**5** 0.2%	**0** 0.0%	**582** 18.2%	**1** 0.0%	99.0% 1.0%
7	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**107** 3.3%	100% 0.0%
	95.0% 5.0%	87.5% 12.5%	81.9% 18.1%	72.8% 27.2%	74.7% 25.3%	99.8% 0.2%	98.2% 1.8%	**85.8%** **14.2%**

Fig. 11. Confusion matrix of the KNN algorithm obtained using 328 features.

Table IX shows the confusion matrix of the binary classification problem resulting from using the KNN algorithm with 328 features. From Table IX, we see that the number of FP equals 0 and the number of FN is 2. The FP rate is equal to 0%, while the FN rate is 1.83%. The accuracy and precision

[12] The contribution of the gravity is equal to $10\ \mathrm{m/s^2}$ along the z-axis of earth-centered coordinate system. In our case, the gravity contribution is measured along the different axes of the accelerometer, which differ from the axes of the earth-centered coordinate system.

TABLE IX
CONFUSION MATRIX OF THE BINARY CLASSIFICATION PROBLEM OF THE KNN ALGORITHM OBTAINED USING 328 FEATURES.

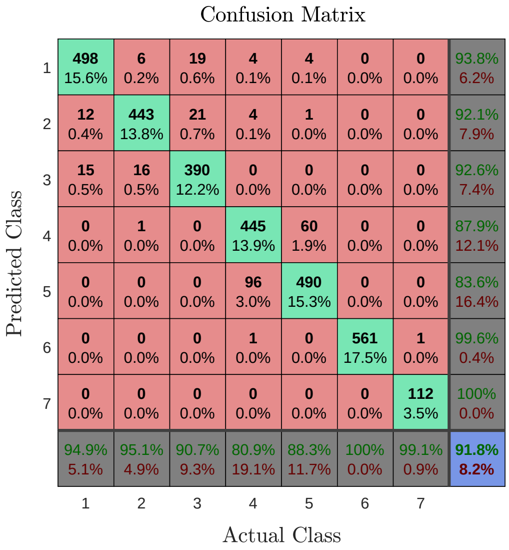
	Actual Non-Fall	Actual Fall
Predicted Non-Fall	3090 **(TN)**	2 **(FN)**
Predicted Fall	0 **(FP)**	107 **(TP)**

TABLE X
CONFUSION MATRIX OF THE BINARY CLASSIFICATION PROBLEM OF THE ANN ALGORITHM OBTAINED USING 328 FEATURES.

	Actual Non-Fall	Actual Fall
Predicted Non-Fall	3087 **(TN)**	1 **(FN)**
Predicted Fall	0 **(FP)**	112 **(TP)**

of fall detection are equal to 98.17% and 100%, respectively.

The confusion matrix of the ANN algorithm in Fig. 12 shows that the ANN algorithm outperforms the KNN algorithm by 6% in terms of overall accuracy. Additionally, the use of 328 features instead of 66 features in conjunction with the ANN algorithm allows improving the classification accuracy by 4%. On the other hand, the increase of the number of features from 66 to 328 results in enhancing the prediction precision for the activities walking, walking upstairs, walking downstairs, and sitting by 9.6%, 7.4%, 6.6%, and 4.1%, respectively. In terms of accuracy, we observe that the use of 328 features instead of 66 features yields an improvement in classification accuracy by 6.5%, 13.8%, 6.6%, and 5.3% for the activities walking, walking upstairs, walking downstairs, and standing, respectively.

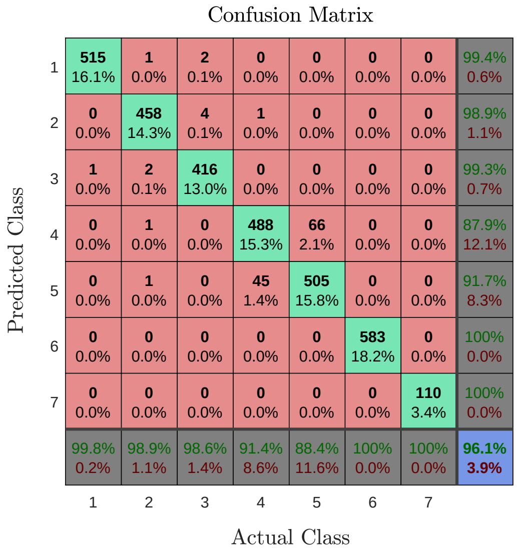


	1	2	3	4	5	6	7	
1	**498** 15.6%	**6** 0.2%	**19** 0.6%	**4** 0.1%	**4** 0.1%	**0** 0.0%	**0** 0.0%	93.8% 6.2%
2	**12** 0.4%	**443** 13.8%	**21** 0.7%	**4** 0.1%	**1** 0.0%	**0** 0.0%	**0** 0.0%	92.1% 7.9%
3	**15** 0.5%	**16** 0.5%	**390** 12.2%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	92.6% 7.4%
4	**0** 0.0%	**1** 0.0%	**0** 0.0%	**445** 13.9%	**60** 1.9%	**0** 0.0%	**0** 0.0%	87.9% 12.1%
5	**0** 0.0%	**0** 0.0%	**0** 0.0%	**96** 3.0%	**490** 15.3%	**0** 0.0%	**0** 0.0%	83.6% 16.4%
6	**0** 0.0%	**0** 0.0%	**0** 0.0%	**1** 0.0%	**0** 0.0%	**561** 17.5%	**1** 0.0%	99.6% 0.4%
7	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**112** 3.5%	100% 0.0%
	94.9% 5.1%	95.1% 4.9%	90.7% 9.3%	80.9% 19.1%	88.3% 11.7%	100% 0.0%	99.1% 0.9%	**91.8%** **8.2%**

Fig. 12. Confusion matrix of the ANN algorithm obtained using 328 features.

Table X provides the confusion matrix of the binary classification problem resulting from using the ANN algorithm with 328 features. From Table X, we see that the number of FP equals 0 and the number of FN is 1. The FP rate is equal to 0%, while the FN rate is 0.88%. The accuracy and precision for fall detection are equal to 99.12% and 100%, respectively.

In Figs. 13 and 14, we provide the confusion matrices for the QSVM and the EBT algorithms, respectively. These two algorithms have a better performance compared to the KNN and the ANN algorithms. The QSVM and the EBT algorithms achieve an overall accuracy of 96.1% and 97.7%, respectively. For the EBT algorithm, the accuracy is 100% for the activities lying and falling, and there is no false alarm for these activities. Moreover, the accuracy and the precision of the EBT algorithm for the activities walking, walking upstairs, and walking downstairs are above 98%. The activities with the lowest accuracy and precision are sitting and standing. The EBT algorithm can classify the activities sitting and standing with an accuracy of 94.6% and 95.4%, respectively. The prediction precisions of the EBT algorithm for sitting and standing are equal to 95.1% and 94.8%, respectively. Note that differentiating between sitting and standing is not very critical.



	1	2	3	4	5	6	7	
1	**515** 16.1%	**1** 0.0%	**2** 0.1%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	99.4% 0.6%
2	**0** 0.0%	**458** 14.3%	**4** 0.1%	**1** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	98.9% 1.1%
3	**1** 0.0%	**2** 0.1%	**416** 13.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	99.3% 0.7%
4	**0** 0.0%	**1** 0.0%	**0** 0.0%	**488** 15.3%	**66** 2.1%	**0** 0.0%	**0** 0.0%	87.9% 12.1%
5	**0** 0.0%	**1** 0.0%	**0** 0.0%	**45** 1.4%	**505** 15.8%	**0** 0.0%	**0** 0.0%	91.7% 8.3%
6	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**583** 18.2%	**0** 0.0%	100% 0.0%
7	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**0** 0.0%	**110** 3.4%	100% 0.0%
	99.8% 0.2%	98.9% 1.1%	98.6% 1.4%	91.4% 8.6%	88.4% 11.6%	100% 0.0%	100% 0.0%	**96.1%** **3.9%**

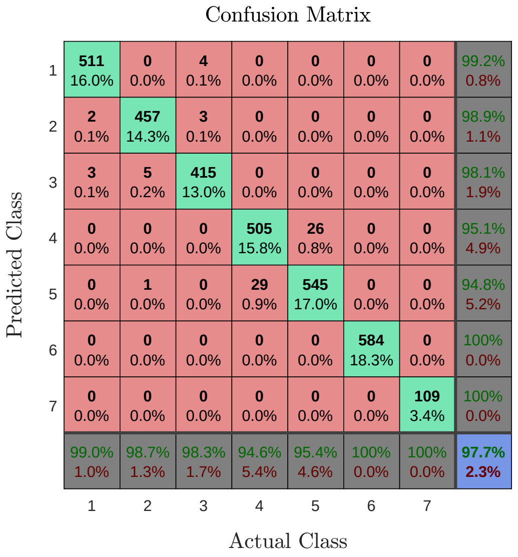
Fig. 13. Confusion matrix of the QSVM algorithm obtained using 328 features.

Table XI illustrates the confusion matrix of the binary classification problem obtained when using the QSVM algorithm with 328 features. From Table XI, we observe that the number of FP equals 0 and the number of FN is 0. The FP rate and FN rate are both equal to 0%, this implies that the fall detection system has zero false alarm and has zero undetected falls.

TABLE XI
CONFUSION MATRIX OF THE BINARY CLASSIFICATION PROBLEM OF THE QSVM ALGORITHM OBTAINED USING 328 FEATURES.

	Actual Non-Fall	Actual Fall
Predicted Non-Fall	3089 **(TN)**	0 **(FN)**
Predicted Fall	0 **(FP)**	110 **(TP)**

Table XII shows the confusion matrix of the binary classifi-



	1	2	3	4	5	6	7	
1	511 16.0%	0 0.0%	4 0.1%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	99.2% 0.8%
2	2 0.1%	457 14.3%	3 0.1%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	98.9% 1.1%
3	3 0.1%	5 0.2%	415 13.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	98.1% 1.9%
4	0 0.0%	0 0.0%	0 0.0%	505 15.8%	26 0.8%	0 0.0%	0 0.0%	95.1% 4.9%
5	0 0.0%	1 0.0%	0 0.0%	29 0.9%	545 17.0%	0 0.0%	0 0.0%	94.8% 5.2%
6	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	584 18.3%	0 0.0%	100% 0.0%
7	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	109 3.4%	100% 0.0%
	99.0% 1.0%	98.7% 1.3%	98.3% 1.7%	94.6% 5.4%	95.4% 4.6%	100% 0.0%	100% 0.0%	97.7% 2.3%

Fig. 14. Confusion matrix of the EBT algorithm obtained using 328 features.

cation problem obtained when using the EBT algorithm with 328 features. From Table XII, we see that the number of FP equals 0 and the number of FN is 0. The FP rate and FN rate are both equal to 0%, thus the fall detection system has an accuracy of 100% and generates zero false alarm.

TABLE XII
CONFUSION MATRIX OF THE BINARY CLASSIFICATION PROBLEM OF THE EBT ALGORITHM OBTAINED USING 328 FEATURES.

	Actual Non-Fall	Actual Fall
Predicted Non-Fall	3090 (**TN**)	0 (**FN**)
Predicted Fall	0 (**FP**)	109 (**TP**)

By comparing the accuracy of the QSVM and the EBT algorithms for different activities, we notice that the EBT algorithm outperforms the QSVM algorithm in classifying the activities sitting and standing, while the QSVM algorithm outperforms the EBT algorithm in terms of accuracy for the activities walking, walking upstairs, and walking downstairs. Both the EBT and the QSVM algorithms reach an accuracy and a precision of 100% in classifying the activities lying and falling. Note that a 100% precision and accuracy for the activity falling is a highly desirable performance. In fact, by achieving a 100% fall detection accuracy, we can build reliable fall detection systems that support the independent living of the elderly, reduce the impact of fall related injuries, and improve the survival rate for persons that experience falls. On the other hand, achieving 100% precision in fall detection means that no false alarm is generated by the algorithm and all detected falls are real falls. Note that if the algorithm generates a false alarm, an ambulance will be sent to the person's house. As the number of false alarms increases, the amount of wasted money increases. Therefore, it is very important to develop a fall detection system without false alarms, which is achieved using the EBT and the QSVM algorithms as shown in Figs. 13 and 14.

V. CONCLUSION

A robust fall detection system is essential to support the independent living of elderlies. In this paper, we have proposed a machine learning approach for fall detection and ADL recognition. We have tested the performance of four algorithms in recognizing the activities falling, walking, walking upstairs, walking downstairs, sitting, standing, and lying based on the acceleration and the angular velocity data. We have proposed new time and frequency domain features and have demonstrated the importance of these features and their positive impact on enhancing the accuracy and precision of the classifier.

Moreover, we have tested the performance of the KNN, ANN, QSVM, and EBT classification algorithms on real-world acceleration data obtained from public databases. The internal parameters of these algorithms have been optimized using the training data. Afterwards, the performance of the trained algorithms has been assessed using the test data. In a first step, only the acceleration data have been used for activity recognition. A feature vector of size 66 has been obtained and has been provided as an input to the classification algorithm. Our results reveal that the KNN, ANN, QSVM, and EBT algorithm achieve an overall accuracy of 81.2%, 87.8%, 93.2%, and 94.1%, respectively.

In a second step, we have extracted new features from both the acceleration and the angular velocity data which has significantly improved the performance of the four classification algorithms. The constructed feature vector has a size of 328. By using the proposed feature vector, we have shown that the KNN, ANN, QSVM, and EBT algorithms achieve an overall accuracy of 85.8%, 91.8%, 96.1%, and 97.7%, respectively. It is worth to mention that the accuracy of fall detection for QSVM and EBT reaches 100% with no false alarm which is the best achievable performance.

ACKNOWLEDGEMENT

This work was supported by the WiCare Project funded by the Research Council of Norway under grant number 261895/F20.

REFERENCES

[1] WHO. (2018, Jun.) World report on ageing and health. [Online]. Available: http://apps.who.int/iris/bitstream/handle/10665/186463/9789240694811_eng.pdf

[2] ——. (2018, Jun.) World Health Organization: Global report on falls prevention in older age. [Online]. Available: https://extranet.who.int/agefriendlyworld/wp-content/uploads/2014/06/WHo-Global-report-on-falls-prevention-in-older-age.pdf

[3] G. Bergen, M. Stevens, and E. Burns. (2018, Jun.) Falls and fall injuries among adults aged $\geq$ 65 years – United States, 2014. [Online]. Available: http://dx.doi.org/10.15585/mmwr.mm6537a2

[4] E. R. Burns, J. A. Stevens, and R. Lee, "The direct costs of fatal and non-fatal falls among older adults – United States," *Journal of Safety Research*, vol. 58, pp. 99–103, Sep. 2016.

[5] R. Igual, C. Medrano, and I. Plaza, "Challenges, issues and trends in fall detection systems," *BioMedical Engineering OnLine*, vol. 12, no. 1, pp. 1–24, Jul. 2013.

[6] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, Bruges, Belgium, Apr. 2013, pp. 24–26.

[7] O. Ojetola, E. Gaura, and J. Brusey, “Data set for fall events and daily activities from inertial sensors,” in *6th ACM Multimedia Systems Conference - MMSys ’15*, Portland, OR, USA, Mar. 2015, pp. 243–248.

[8] A. Wertner, P. Czech, and V. Pammer-Schindler, “An open labelled dataset for mobile phone sensing based fall detection,” in *12th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MOBIQUITOUS 2015)*, Coimbra, Portugal, Jul. 2015, pp. 277–278.

[9] A. Sucerquia, J. D. López, and J. F. Vargas-Bonilla, “SisFall: A fall and movement dataset,” *Sensors*, vol. 17, no. 1, pp. 1–14, Jan. 2017.

[10] E. Casilari, J. A. Santoyo-Ramón, and J. M. Cano-García, “Analysis of a smartphone-based architecture with multiple mobility sensors for fall detection,” *PLoS ONE*, vol. 11, pp. 1–17, Dec. 2016.

[11] O. P. Popoola and K. Wang, “Video-based abnormal human behavior recognition–A review,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 865–878, Nov. 2012.

[12] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, “Robust video surveillance for fall detection based on human shape deformation,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 5, pp. 611–622, May 2011.

[13] C. Zhang, Y. Tian, and E. Capezuti, “Privacy preserving automatic fall detection for elderly using RGBD cameras,” in *International Conference on Computers for Handicapped Persons (ICCHP 2012)*. Linz, Austria: Springer, Berlin, Heidelberg, Jul. 2012, pp. 625–633.

[14] I. Charfi, J. Miteran, J. Dubois, M. Atri, and R. Tourki, “Definition and performance evaluation of a robust SVM based fall detection solution,” in *2012 Eighth International Conference on Signal Image Technology and Internet Based Systems*. Naples, Italy: IEEE, Nov. 2012, pp. 218–224.

[15] H. A. Nguyen and J. Meunier, “Gait analysis from video: camcorders vs. Kinect,” in *International Conference Image Analysis and Recognition (ICIAR 2014)*. Vilamoura, Portugal: Springer, Oct. 2014, pp. 66–73.

[16] R. K. Tripathy, L. N. Sharma, S. Dandapat, B. Vanrumste, and T. Croonenborghs, “Bridging the gap between real-life data and simulated data by providing a highly realistic fall dataset for evaluating camera-based fall detection algorithms,” *Healthcare Technology Letters*, vol. 3, no. 1, pp. 6–11, Mar. 2016.

[17] K. Sehairi, F. Chouireb, and J. Meunier, “Comparative study of motion detection methods for video surveillance systems,” *Journal of Electronic Imaging*, vol. 26, no. 2, pp. 26–29, Apr. 2017.

[18] J. Klonovs *et al.*, *Distributed Computing and Monitoring Technologies for Older Patients*, 1st ed. London, UK: SpringerBriefs in Computer Science, 2016.

[19] A. Bourke, P. Van de Ven, A. Chaya, G. OLaighin, and J. Nelson, “Testing of a long-term fall detection system incorporated into a custom vest for the elderly,” in *30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS 2008)*, Vancouver, BC, Canada, Aug. 2008, pp. 2844–2847.

[20] P. Barralon, I. Dorronsoro, and E. Hernandez, “Automatic fall detection: Complementary devices for a better fall monitoring coverage,” in *IEEE 15th International Conference on e-Health Networking, Applications and Services (Healthcom 2013)*, Lisbon, Portugal, Oct. 2013, pp. 590–593.

[21] P. Kostopoulos, T. Nunes, K. Salvi, M. Deriaz, and J. Torrent, “F2D: A fall detection system tested with real data from daily life of elderly people,” in *17th International Conference on e-Health Networking, Application & Services (HealthCom)*, Boston, MA, USA, Oct. 2015, pp. 397–403.

[22] O. Ojetola, E. I. Gaura, and J. Brusey, “Fall detection with wearable sensors–Safe (smart fall detection),” in *Seventh International Conference on Intelligent Environments*, Jul. 2011, pp. 318–321.

[23] P. Putra, J. Brusey, and E. Gaura, “A cascade-classifier approach for fall detection,” in *5th EAI International Conference on Wireless Mobile Communication and Healthcare (MOBIHEALTH’15)*, London, UK, Oct. 2015, pp. 94–99.

[24] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, “Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine,” in *International Workshop of Ambient Assited Living*, Vitoria-Gasteiz, Spain, Dec. 2012, pp. 216–223.

[25] A. B. Williams and F. J. Taylor, *Electronic Filter Design Handbook*, 4th ed. New York, USA: McGraw-Hill, 2006.

[26] J.-Y. Yang, J.-S. Wang, and Y.-P. Chen, “Using acceleration measurements for activity recognition: An effective learning algorithm for constructing neural classifiers,” *Pattern Recognition Letters*, vol. 29, no. 16, pp. 2213–2220, 2008.

[27] A. M. Khan, Y.-K. Lee, S. Y. Lee, and T.-S. Kim, “Human activity recognition via an accelerometer-enabled-smartphone using kernel discriminant analysis,” in *2010 5th International Conference on Future Information Technology*, Busan, South Korea, May 2010, pp. 1–6.

[28] C. M. Bishop, *Pattern Recognition and Machine Learning*, 1st ed. Cambridge, UK: Springer, 2006.

[29] I. Steinwart and A. Christmann, *Support Vector Machines*, 1st ed. New York, USA: Springer, 2008.

[30] T. G. Dietterich, “An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization,” *Machine Learning*, vol. 40, no. 2, pp. 139–157, Aug. 2000.

[31] R. W. Schafer, “What is a Savitzky-Golay filter? [Lecture notes],” *IEEE Signal Processing Magazine*, vol. 28, no. 4, pp. 111–117, Jul. 2011.

[32] D. Bailey and P. Swarztrauber, “A fast method for the numerical evaluation of continuous Fourier and Laplace transforms,” *SIAM Journal on Scientific Computing*, vol. 15, no. 5, pp. 1105–1110, Sep. 1994.

[33] J. M. Knudsen and P. G. Hjorth, *Elements of Newtonian Mechanics: Including Nonlinear Dynamics*, 3rd ed. Springer-Verlag Berlin Heidelberg, 2000.

[34] M. Vallejo, C. Isaza, and J. López, “Artificial Neural Networks as an alternative to traditional fall detection methods,” in *35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Osaka, Japan, Jul. 2013, pp. 1648–1651.

[35] A. Vellido, J. D. M. n Guerrero, and P. J. G. Lisboa, “Making machine learning models interpretable,” in *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2012)*, Bruges, Belgium, Apr. 2012, pp. 163–172.

[36] W. Samek, T. Wiegand, and K. Müller, “Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models,” *arXiv e-prints, arXiv:1708.08296*, Aug. 2017.